DEVELOPMENT OF A ROBUST OPTICAL IMAGE REGISTRATION ALGORITHM FOR NEGATING SPECKLE NOISE EFFECTS IN COHERENT IMAGES GENERATED BY A LASER IMAGING SYSTEM

THESIS

Darren R. Sabo, Major, USAF

AFIT/GE/ENG/05-17

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or U.S. Government.

AFIT/GE/ENG/05-17

DEVELOPMENT OF A ROBUST OPTICAL IMAGE REGISTRATION ALGORITHM FOR NEGATING SPECKLE NOISE EFFECTS IN COHERENT IMAGES GENERATED BY A LASER IMAGING SYSTEM

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Darren R. Sabo, B.S.E.E.

Major, USAF

March 2005

AFIT/GE/ENG/05-17

DEVELOPMENT OF A ROBUST OPTICAL IMAGE REGISTRATION ALGORITHM FOR NEGATING SPECKLE NOISE EFFECTS IN COHERENT IMAGES GENERATED BY A LASER IMAGING SYSTEM

Darren R. Sabo, B.S.E.E.
Major, USAF

Approved:

/signed/	10 Mar 2005
Dr. Stephen C. Cain, (Chairman)	date
/signed/	10 Mar 2005
Dr. Steven C. Gustafson, (Member)	date
/signed/	10 Mar 2005
Maj Matthew E. Goda, PhD, (Member)	date

Abstract

The Air Force Research Laboratory (AFRL) Sensors Directorate has constructed and tested a coherent LIght Detection And Ranging (LIDAR) imaging system called Laservision. Registration of individual images remains a significant problem in the generation of useful images collected using coherent imaging systems.

Coherent images typically contain significant speckle noise created by the coherent nature of the laser. Each image collected by the system must be properly registered to allow for averaging the images to produce a single image with adequate resolution to allow detection and identification algorithms to operate accurately or for system operators to perform target detection and identification within a scene.

An investigation of the performance of a new image registration algorithm designed using laser speckle noise statistics is conducted on data collected from the Laservision system. This thesis documents the design and performance of the proposed technique compared to that of a standard cross-correlation algorithm.

Based on using only speckle noise statistics, the simulated data test results indicate that there is a small range of low average signal-to-noise ratios (SNR) where there is the potential to improve the shift estimation error by 0.1 to 0.16 pixels.

Acknowledgements

First and foremost, I owe my greatest acknowledgement to my Lord and Savior, Jesus Christ, without whom I'd be lost in this world, seeking my own gains by my own means. He has directed my path to this institution, and by His grace and providence gave me the ability to finish this journey. Though He may not care about the work I've done to support the Air Force, my prayer has been that I would glorify and honor Him in all my thoughts, words and actions, to include my efforts and performance in completing this masters program. Whatever good has resulted from my research is for His glory, and I give Him the praise and honor for allowing me to be His servant.

! SDG - Soli Deo Gloria: J. S. Bach appended these initials at the end of each of his Cantatas scores. Soli Deo Gloria, to the Glory of God alone. They signified his deep devotion and his desire to serve God through his music.

The Lord also has blessed me with a tremendous amount of support during my tenure at AFIT. The kind of support that only comes from a loving God, who longs to bless those who answer His call and serve Him. I would be greatly remiss if I did not acknowledge those He has placed in my life to lift and encourage me.

To my wife: I thank you for your constant love, support and encouragement when the days seemed to never end. God has truly blessed me with the perfect partner to complete me and be the sunshine of my days and the moonlight of my nights. I love you more with each day the Lord blesses me with you.

To my daughters: I thank you my little angels for being the joy of my heart, keeping me true to what's most important (family), and always making sure we have family night each week. Thank you for your patience and forgiveness as Daddy worked the long hours to finish this work. I love you more with each day the Lord blesses me with you.

To my Zoo Brothers in Christ: Without you I would not have survived academically through the first two quarters (thank you C-dawg), and I thank you for the

support and encouragement to continue to trust in the Lord and just keep trying my best.

I also thank the Lord for blessing me with a great advisor, Dr Stephen Cain. Thank you Dr Cain for your patient guidance, tutoring the complexity of speckle noise and coherency, and encouragement that I could accomplish more than I thought. To my committee, Dr Steven Gustafson and Major Matt Goda, thank you for your time and support, and providing the input needed to make my thesis a better document.

In conclusion, I leave the future students of AFIT with this to ponder:

> ! The great German scientist and mathematician, Johann Kepler, discoverer of the three laws for planetary motion that are the basis for our understanding of the solar system today, clearly recognized God's design in this world. He felt that it was our responsibility to discover its complexities; the scientist's duty was to discover what mathematical formula God had used. Through seeing His greatness in creation he thought we would deepen our worship of Him [3]. Have we?

At the end of his life Kepler's prayer was: "I give you thanks, Creator and God, that you have given me this joy in thy creation, and I rejoice in the works of your hands. See I have now completed the work to which I was called. In it I have used all the talents you have lent to my spirit" [3].

I pray I have done the same with my time at AFIT.

"Trust in the Lord with all your heart and lean not on your own understanding; in all your ways acknowledge Him, and He will make your paths straight. Do not be wise in your own eyes; fear the LORD and shun evil. This will bring health to your body and nourishment to your bones." Proverbs 3:5-8

Darren R. Sabo

Table of Contents

Page

List of Figures

Figure Page

List of Abbreviations

Abbreviation		Page
AFRL	Air Force Research Laboratory	iv
LIDAR	LIght Detection and Ranging	iv
SNR	signal-to-noise ratio	iv
LADAR	Laser Radar	1
DARPA	Defense Advance Research Programs Agency	1
SPI-3D	Standoff Precision ID in 3-D	1
TCT	time-critical-targeting	4
MSE	mean-square-error	5
LITE	Laser-illuminated Image Translation Estimation	6
ML	maximum likelihood	6
FFT	fast Fourier transform	7
CCD	charge-coupled device	12
FOV	field of view	14
PDF	probability density function	23

DEVELOPMENT OF A ROBUST OPTICAL IMAGE REGISTRATION ALGORITHM FOR NEGATING SPECKLE NOISE EFFECTS IN COHERENT IMAGES GENERATED BY A LASER IMAGING SYSTEM

I. Introduction

This thesis describes the development of a robust optical image registration algorithm for negating speckle noise effects in coherent images generated by a laser imaging system. Current registration algorithms are analyzed based on performance metrics as applied in a LIght Detection And Ranging (LIDAR) or laser radar (LADAR) imaging system, and a new algorithm is designed specifically for LADAR image registration. Research and analysis is conducted on image data and image registration algorithms are implemented in MATLAB®, with the goal of developing a registration strategy superior to that currently used in the Air Force Research Laboratory (AFRL) Laservision system. This image registration research also serves as an enabling technology for the Defense Advance Research Programs Agency (DARPA) Standoff Precision ID in 3-D (SPI-3D) program.

This chapter outlines the problem statement, illustrates the benefit of properly registering images, and concludes with the methodology and development behind the thesis.

1.1 Problem Statement

Coherent images generated by LADAR imaging systems typically contain high levels of speckle noise created by the "coherent nature of the collected energy" of the laser beam [5]. Coherent images are those where the phase components of the laser light reflections remain relatively constant, or retain a definite relationship, while the

image is being captured by the system. A detailed explanation of the phenomenon of speckle noise is provided in Chapter II.

Each image collected by the system must be properly registered to a baseline image before a set of images can be averaged. Averaging serves to suppress the speckle noise and produce a single image with adequate resolution to allow detection and identification algorithms to operate accurately or for system operators to perform target detection and identification within a scene. Additionally, atmospheric turbulence and illuminator motion generate a spatial shift in each image which further complicates the registration process. The current registration algorithm used in the Laservision system could be improved to yield better imaging performance.

Alternative registration algorithms currently available either do not perform well within a laser imaging environment, are too computer resource intensive for the Laservision imaging system, or commonly assume a normal (Gaussian) noise distribution in estimating the motion parameters for the registration algorithm, which contributes to lower performance. The Gaussian noise assumption, while frequently used as a safe representation when a distribution is not known, does not characterize the effects of coherent image speckle noise on the observed image and therefore is not the best estimate for the motion parameters used in the registration algorithm. Hence, while there is some image resolution improvement using current registration algorithms, a new algorithm designed using a better characterization of the speckle noise effect on the observation distribution is a potential solution for image registration within a LADAR system. Based on Goodman's research of photon count statistics for polarized thermal light and an arbitrary counting interval, the observed distribution is known as a negative binomial distribution [4]. This research effort develops a new registration estimation algorithm using a negative binomial realization of the noise.

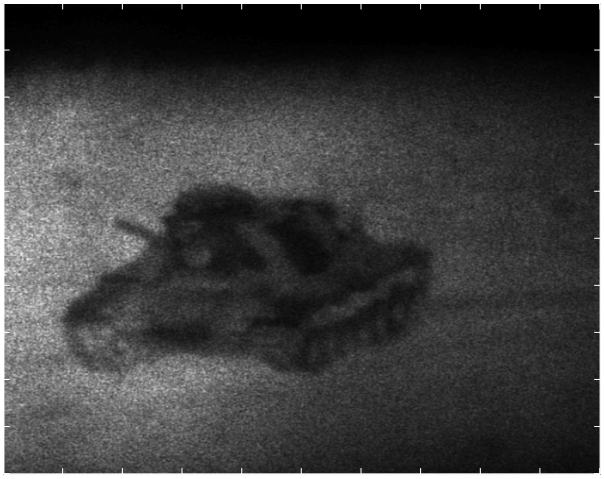


Figure 1.1: **Speckle Noise Illustration.** Unprocessed image from the AFRL Laservision system highlighting the speckle noise evident in individual images (i.e., "graininess" or "fuzziness").

1.2 Benefit of Proper Registration

Referring to Figures 1.1, 1.2, 1.3, the obvious benefit of properly registering a set of images can be seen. Figure 1.1 represents an unprocessed image from the AFRL Laservision system that highlights the speckle noise evident in individual images (i.e., "graininess" or "fuzziness"). Figures 1.2 and 1.3 illustrate the effects of averaging 10 frames of data without registration and with registration, respectively. Note in Figure 1.2 that the averaging process suppresses some of the speckle noise, but the lack of registration, or aligning the images, severely blurs the image. However, Figure 1.3 demonstrates the power of registration coupled with the averaging process, which drastically suppresses the speckle noise effect, thus creating a much clearer image for further processing. It is important to note that the laser images were taken with a stationary system with very little movement. Hence, the majority of the x and y (translational) estimated by a registration algorithm are shift errors induced by the speckle noise. It is evident that a significant image recognition improvement is obtained with the averaging process, but only if the image is properly registered [5]. Therefore, coherent laser imaging systems require accurate and efficient image regis-

tration algorithms as a key enabling technology to meet operator target recognition requirements that improve time-critical-targeting (TCT) capabilities.

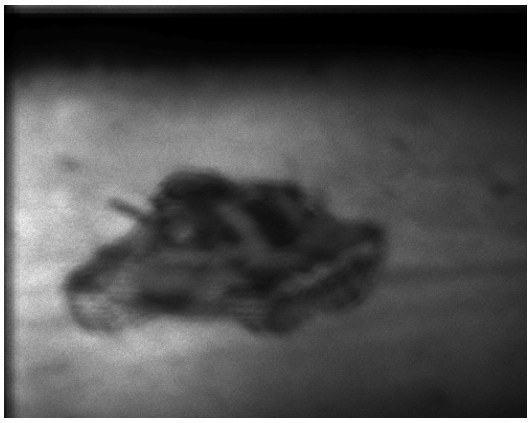


Figure 1.2: **Temporal Averaging without Registration Illustration.** This figure highlights the effects of averaging 10 frames of data without registration (or image alignment). The speckle noise effect is suppressed to an extent, but the lack of registration causes severe blurring of the image.

Figure 1.3: **Registration and Temporal Averaging Illustration.** This figure highlights the effects of averaging 10 frames of data with proper registration, or image alignment. The speckle noise effect is significantly suppressed creating a much clearer image for further processing.

1.3 Methodology and Development Outline

The concept of image registration tailored for coherent laser imaging systems is a recent development with limited documentation and consequently limited research. Furthermore, initial research on registration algorithms considered for use in the Laservision system indicates only two out of eight image registration algorithms analyzed by [5] show any promise of both adequate error and time performance metrics. Analysis conducted by [5] on these two registration algorithms uses mean-square-error (MSE) and computer processing time performance for the metric comparison. AFRL/SNJT provided real-world ground-test laser image data for performing the research and analysis. A comparison and analysis study is conducted using one of the top two image registration algorithms analyzed by [5] and derived and implemented in MATLAB® versus the new algorithm designed in this research effort, also derived and implemented in MATLAB®.

This thesis provides a brief review of the two registration algorithms in [5] that produced the lowest error and time metrics, and it highlights strengths and weaknesses as applied to a laser imaging system. From this background it is evident that the development of a new algorithm that uses the true statistical characterization of the speckle noise effect should produce more accurate registration for a laser imaging system.

The fundamental design methodology incorporates an estimation theory approach described by Van Trees [8] to create a new registration algorithm tailored for use in a laser imaging system. This approach begins by considering image registration as a fundamental problem of estimating motion between scenes, as pointed out in [7]. This theoretical approach is covered in detail in Chapter II.

The scope of this thesis is limited to the development of a new registration algorithm design that registers images with regard to only translational shifts (e.g. only those shifts in the vertical and horizontal direction). Hence, the algorithm designed

is called Laser-illuminated Translation Estimation (LITE) registration, and will be referred to as such throughout the remainder of this document.

The results from the LITE registration analysis are presented in Chapter IV. This thesis incorporates both a maximum likelihood (ML) estimator that measures the estimation process efficiency, as well as a standard deviation error metric on the two algorithms being compared. Additionally, the processing time performance metric analyzed by [5] is considered. Performance verification of the LITE algorithm is conducted on test images with simulated speckle noise versus one of the top two algorithms analyzed by [5], the cross-correlation algorithm, using registration standard deviation error measured in pixels over varying signal-to-noise ratios (SNR) (e.g. light intensity levels) as the performance metric. Further test analysis comparing these same two algorithms is also conducted using real-world ground test images containing the actual speckle noise effects of the Laservision imaging system. The cross-correlation algorithm is selected due to its ease of implementation in MATLAB®, and more importantly it can be shown that cross-correlation is the best shift estimator for a signal with white Gaussian noise. This is clarified in the Background section of Chapter II and proved in Appendix A.

Finally, this thesis presents conclusions on the LITE registration algorithm design and proposes avenues for future research in laser image registration, as well as other image and signal environments that could benefit from this new shift estimation design.

II. Background

This chapter provides a brief description of past work accomplished in laser image processing. It includes a brief review of the two registration algorithms in [5] that produce both the fastest processing times and the lowest mean-square-error (MSE), and it highlights their strengths and weaknesses as applied to a laser imaging system. It continues with a brief system description, a short tutorial on how speckle is created, a short tutorial on image registration, and description of the data model used in this thesis.

2.1 Laser Image Registration Historical Research

The concept of image registration tailored for coherent laser imaging systems is very new, and consequently documentation and research in this image processing discipline is limited. As such, there is currently no known research or development of a registration algorithm designed using speckle noise statistics to improve registration techniques for laser imaging systems. Therefore, the background research is limited to a brief review of the top two image registration algorithms analyzed by [5] that show the most promise based on their processing times and error performance, and it highlights their less than optimum applicability to this problem. These candidate registration algorithms are cross-correlation and vector-block.

2.1.1 Cross-correlation. Cross-correlation, also referred to as the phase-correlation method, uses fast Fourier transforms (FFT) to calculate a correlation peak between two identical but spatially shifted images [5]. This peak is located at pixel offsets in both the x and y direction as compared to the autocorrelation peak of the baseline image [5]. Mathematically, the cross-correlation method is derived through the following steps:

Consider two true images i_1 and i_2 of the same scene. They are described by

$$i_1(x, y) = i_2(x - \alpha, y - \beta), \tag{2.1}$$

where i_2 is a shifted copy of the true image i_1.

The cross-correlation between i_1 and i_2 is

$$C_{\alpha,\beta}(z,w) = \sum_{x=1}^{N}\sum_{y=1}^{M} d_2(x,y)d_1(x+z,y+w), \tag{2.2}$$

where d_1 and d_2 are the observations of the true images i_1 and i_2 plus their respective noise components n_1 and n_2, or

$$d_1(x,y) = i_1(x,y) + n_1(x,y), \tag{2.3}$$

$$d_2(x,y) = i_1(x-\alpha, y-\beta) + n_2(x,y). \tag{2.4}$$

Using 2.3 and 2.4, the role of the noise components is analyzed by taking the Fourier transform of $C_{\alpha,\beta}$

$$\begin{aligned} F\{C_{\alpha,\beta}(z,w)\} &= D_1(f_x,f_y)D_2^*(f_x,f_y) \\ &= [I_1(f_x,f_y) + N_1(f_x,f_y)][I_2^*(f_x,f_y) + N_2^*(f_x,f_y)] \\ &= I_2^*(f_x,f_y)I_2(f_x,f_y)e^{-j2\pi(f_x\alpha+f_y\beta)} + N_1(f_x,f_y)I_2^*(f_x,f_y) \\ &\quad + I_1(f_x,f_y)N_2^*(f_x,f_y) + N_1(f_x,f_y)N_2^*(f_x,f_y), \end{aligned} \tag{2.5}$$

where I_1 is the shifted version of I_2 with the shift captured in the linear phase $e^{-j2\pi(f_x\alpha+f_y\beta)}$ in the frequency domain.

The inverse Fourier transform isolates the α and β shifts:

$$\begin{aligned} F^{-1}\{D_1(f_x,f_y)D_2^*(f_x,f_y)\} &= R_{ii}(z-\alpha,w-\beta)+R_{n_1i_2}(z,w,\alpha,\beta) \\ &\quad +R_{n_2i_1}(z,w,\alpha,\beta)+R_{n_1n_2}(z,w,\alpha,\beta), \end{aligned} \tag{2.6}$$

$$\begin{aligned} C_{\alpha,\beta}(z,w) &= R_{ii}(z-\alpha,w-\beta)+R_{n_1i_2}(z,w,\alpha,\beta) \\ &\quad +R_{n_2i_1}(z,w,\alpha,\beta)+R_{n_1n_2}(z,w,\alpha,\beta). \end{aligned} \tag{2.7}$$

Since the the image and the noise are assumed to be uncorrelated, and the noise from frame to frame is also assumed to be uncorrelated, the last three terms go to zero; thus

$$C_{\alpha,\beta}(z,w)=R_{ii}(z-\alpha,w-\beta). \tag{2.8}$$

The cross-correlation method used in the registered image of Figure 1.3 is "generally regarded as a robust estimator in the presence of noise" [2]. The drawback to using this method is that the FFTs require more computer resources than the Laservision system can allocate for this image processing task. Additionally, it can be shown (Appendix A) that cross-correlation is the best estimator for Gaussian noise, which as mentioned earlier is not the optimum distribution model for characterizing the speckle noise effect on the observed image photon count distribution.

2.1.2 Vector-block. The vector-block method converts the image matrix into two vector sums along the x and y axis by summing the pixels along the horizontal and vertical planes. The two vectors are then compared to the baseline image vectors to determine the shift between the images. This procedure is depicted graphically in Figure 2.1, where the column vector $d_1^y(x,y)$ is the sum of the pixels in the image across the columns and the row vector $d_1^x(x,y)$ is the sum of the pixels in the image across the rows. The algorithm uses these two vectors in the cross-correlation registration

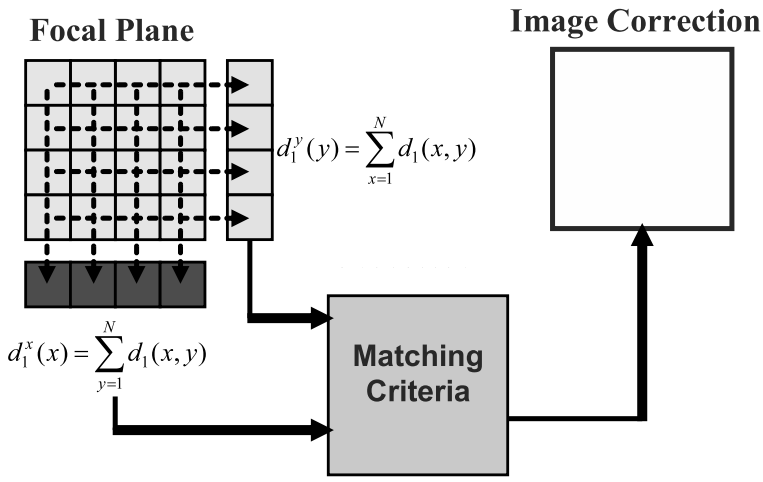


Figure 2.1: **Vector-Block Registration Method Illustration.** The vector-block method converts the image matrix into two vector sums along the x and y axis by summing the pixels along the horizontal and vertical planes. The two vectors are then compared to the baseline image vectors to determine the shift between the images.

method, versus taking the cross-correlation for each pixel in the matrix. The cross-correlation method is derived as before using these vectors through the following steps:

Consider two true images i_1 and i_2 of the same scene. They are

$$i_{x1}(n) = \sum_{m=0}^{M-1} i_1(n,m) = d_x^1, \tag{2.9}$$

$$i_{y1}(m) = \sum_{n=0}^{N-1} i_1(n,m) = d_y^1, \tag{2.10}$$

where d_x^1 and d_y^1 are the row and column vectors from Figure 2.1.

Then the two images are

$$i_1(n,m) = i_2(n - \triangle_x, m - \triangle_y), \tag{2.11}$$

where i_2 is a shifted copy of the true image i_1, and $\triangle_x$ and $\triangle_y$ are the shift values of the respective vectors.

Similarly for the second image

$$i_{x2}(n) = \sum_{m=0}^{M-1} i_2(n, m). \tag{2.12}$$

Continuing with just the row vector yields

$$\begin{aligned} i_{x1}(n) &= \sum_{m=0}^{M-1} i_2(n - \triangle_x, m - \triangle_y) \\ &= i_{x2}(n - \triangle_x) \end{aligned} \tag{2.13}$$

where i_{x2} is simply a shifted row vector version of i_{x1}.

The cross-correlation of the row vectors is

$$\begin{aligned} C_x(z) &= \sum_{n=\triangle_x}^{N-\triangle_x} i_{x2}(n) i_{x1}(z + n) \\ &= \sum_{n=\triangle_{xmax}}^{N-\triangle_{xmax}} i_{x2}(n) i_{x2}(z + n - \triangle_x) \end{aligned} \tag{2.14}$$

and is similar to the 2-D cross-correlation done previously. The Fourier transform of Equation 2.14 is

$$F\{C_x(z)\} = I^*_{x2}(f_x) I_{x2}(f_x) e^{-j2\pi f_x \triangle_x}, \tag{2.15}$$

and the inverse Fourier transform is

$$C_x(z) = R_{x2}(z - \triangle_x) \tag{2.16}$$

This technique, while losing some accuracy, reduces the cross-correlation computational load due to "reducing the dimensionality of the calculations from N^2 to 2 x N, where N is the number of pixels on a side of a square image" [5]. However, while it may prove useable in the Laservision System, the fundamental problem of the distribution model for the speckle noise effect on the observed image still exists for this algorithm.

2.2 Cross-correlation and Vector-block Analysis

A self-study analysis is conducted on the two candidate registration algorithms, where simulations were performed to evaluate and compare their MSE and processing time performance metrics as done by [5]. A detailed description of the simulation methodology and analysis and results is provided in Appendix B.

2.3 System Description

The Laservision system illuminates a scene with a laser beam, then receives the reflected laser energy. It uses a "moderate-power laser focused to illuminate targets at 3 and 10 kilometers. The system receives reflected coherent light from the laser using an optical telescope which feeds a charge-coupled device (CCD) detector to collect the scene intensity" [5]. The receiver is designed to capture ten images per second from the target scene. Figure 2.2 illustrates the basic system concept of the Laservision system. The actual image processing architecture is captured in the block diagram of Figure 2.3, where the incoming images may be noise pre-filtered before processing by the image registration block where the motion parameters (α and β shifts) are calculated. Once the motion parameters are determined, the images are passed to the next block where the motion compensation is executed to align the images based on the shifts. After alignment the set of images are temporally averaged to output a single noise-reduced and clearer image for further processing. The AFRL provided a large data base of 300 images of coherent reflected laser data at a range of 3 kilometers generated by the Laservision system. These target images consisted of

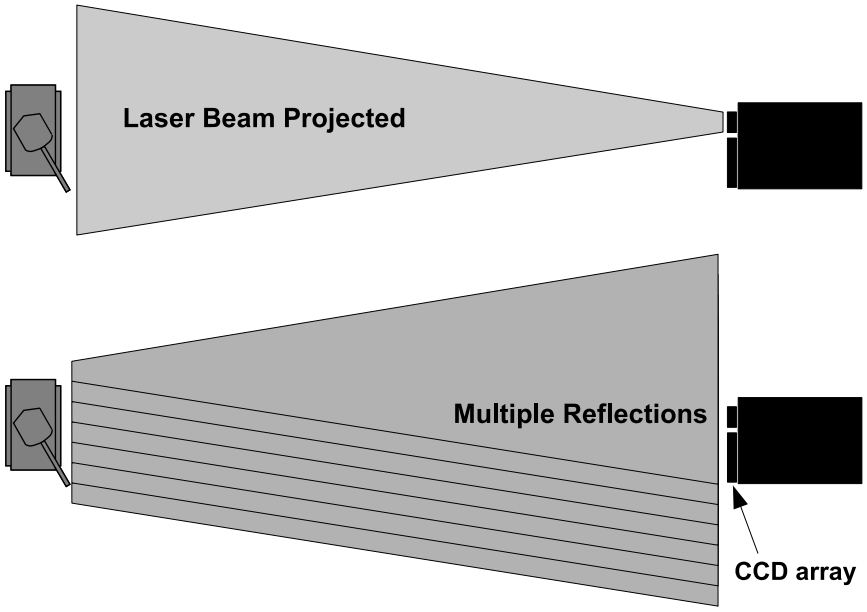


Figure 2.2: **Basic Laservision System Concept.** The AFRL Laservision laser imaging system uses a moderate-power laser focused to illuminate targets at 3 and 10 kilometers. The system receives reflected coherent light from the laser using an optical telescope which feeds a CCD detector.

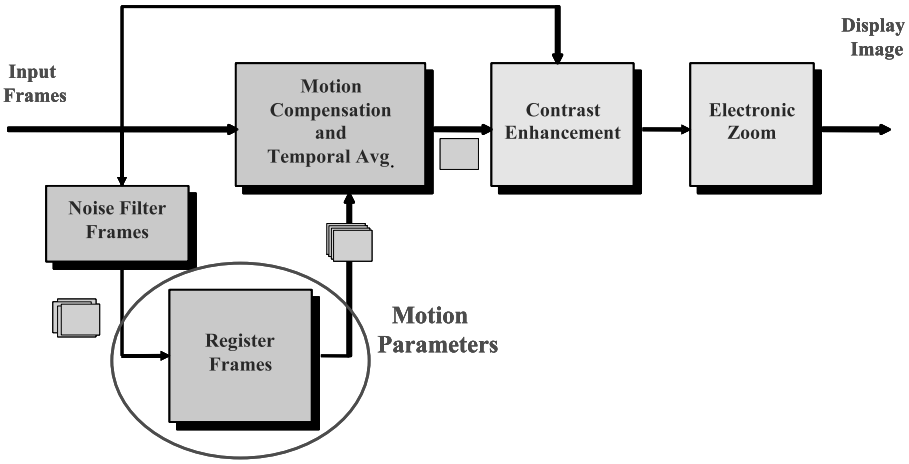


Figure 2.3: **Image Signal Processing Architecture.** A notional example of signal processing executed within a laser image system. This example illustrates registering, or aligning, a set of images, then temporally averaging them to create a single noise-suppressed image for further processing.

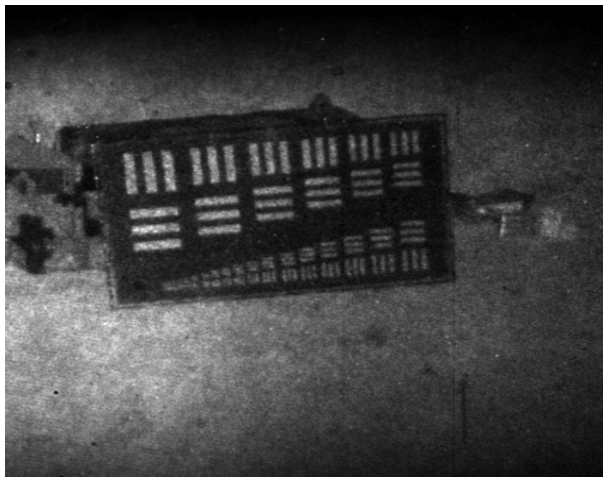


Figure 2.4: **Truck with Resolution board at 3 km.** Example image from the large laser image data base provided by the AFRL.

various vehicles, including trucks with resolution boards mounted on the side and a tank. Figures 2.4 and 2.5 are representative images from the data set that illustrate the speckle noise generated by the coherency of the collected energy from the laser illuminator.

2.4 Coherency and Speckle Noise

To help understand the phenomenon of speckle noise and how coherent laser images create this effect, a brief explanation is offered. Referring to Figures 2.2 and 2.6, a very simplistic scenario demonstrates how speckle is generated in the image.

Recall from the basic system concept illustration in Figure 2.2 that the laser beam is reflected back at the detector by all the various surfaces within the field of view (FOV), thus creating multiple light wave reflections. Recall also that the phase of the projected light wave is assumed constant while the detector is capturing a particular scene, and that the Laservision system is designed to capture ten images per second. However, although the phase is relatively constant for one pulse of the laser beam light (e.g., the scene capture interval; 1μ-second for Laservision), it can

Figure 2.5: **Tank at 3 km.** Example image from the large laser image data base provided by AFRL.

be changed by the roughness of the surfaces within the image scene. This variation in surface roughness can be as little as $1\mu m$ and still create a phase change. A further complication is that the wavelength of the laser beam is coincidentally on the order of $1\mu m$ as well.

There may be thousands of reflected waves during a capture period, but to demonstrate how speckle noise is created in a laser generated image, consider only two points in the scene reflecting only two waves back to the CCD detector. Assume the variation between the two surface points is at least $1\mu m$. Also, consider only two of the pixels in the CCD detector and examine the effect on those two pixels for two extreme cases of the level of light intensity received by each pixel.

Figure 2.6 highlights these two extreme cases, the first of which shows the respective light waves reflected from points one and two arriving perfectly out of phase at CCD pixel two. The effect is a cancellation of the light intensity level registered in the CCD pixel, thus creating a darker pixel point in the observed image scene than the true single point reflection energy level. Similarly, the other extreme is when the light waves arrive perfectly in phase at pixel one. The effect in this case is additive

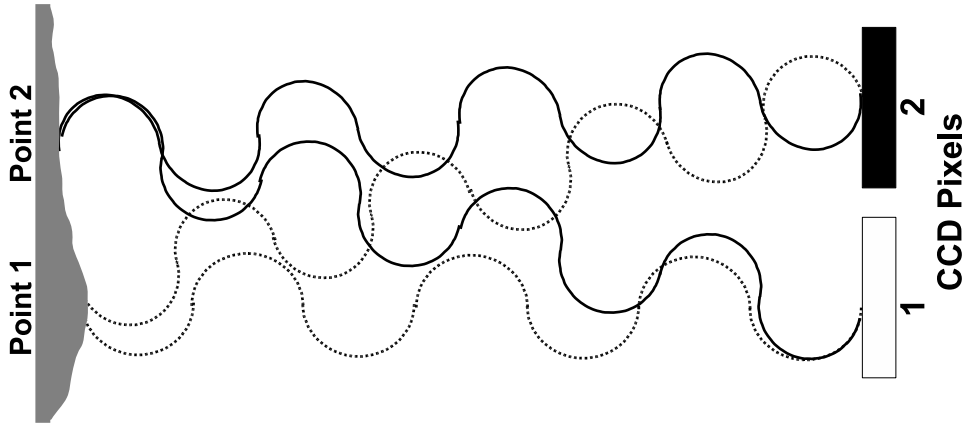


Figure 2.6: **Simplistic Illustration of Phase Coherency and Speckle Noise.** Surface roughness can cause the phase of the laser light to shift, creating varying light intensities as the reflected light is captured by the CCD, thus creating speckle noise in the observed image.

and the light intensity level is increased; thus the CCD pixel registers a brighter point in the observed scene than the true single point reflection energy level.

From this simplistic example of the reflected returns for the extreme phase arrival cases, the multiplicative nature of the observed phenomenon is clearly evident. The multiple variations of phase arrivals on the CCD detector from the thousands of energy returns from the image scene naturally cause varying light intensities on the true image single point reflections, thus blurring or "speckling" the image with noise.

2.5 Image Registration

Image registration is typically defined as a process that compares two or more images by calculating differences in translation (horizontal and vertical shifts), rotation, and scaling in order to match or align the images for further processing [6]. Registration is therefore a key initial image processing step before any target detection techniques can be implemented. To understand the basics of image registration, four essential elements must be defined as outlined by [1]: feature space, search space, search strategy, and similarity metric.

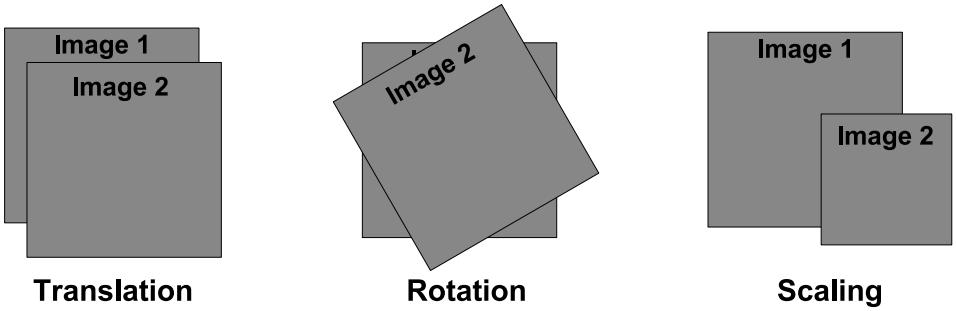


Figure 2.7: **Search Space Transformations.** Translation, rotation, and scaling are the common transformations used in aligning images for registration.

2.5.1 Feature Space. According to Brown [1], feature space is defined as a set of features representative of the image for matching purposes. Examples include pixel intensity, image contours, edges, and line intersections [6]. Appropriate feature selection is the foundation for developing an accurate registration algorithm [1]. For the LITE registration algorithm the feature space is the pixel light intensity, since the actual photon count per pixel is the observed image distribution.

2.5.2 Search Space. Search space defines the transformations (translation, rotation, scaling) that are used in aligning the images during registration [1]. Figure 2.7 illustrates these three common transformation types.

2.5.3 Search Strategy. Search strategy is the process of choosing the set of transformations from the search space that are used in registering consecutive images [1]. As noted previously, in this thesis the development of the LITE registration algorithm is limited to registering images with regard to only translational shifts (i.e., only those shifts in the vertical and horizontal direction).

2.5.4 Similarity Metric. According to Brown, the similarity metric in the context of image registration is a measurement used to determine how well the two images match [1]. Brown cites some similarity metrics as cross-correlation, statistical correlation and matched filters, and Sum of Absolute Differences of Intensity. This

measurement technique is the heart of the registration algorithm, and is the "engine" that determines how the images are aligned. The log-likelihood function used in the LITE registration algorithm developed in this thesis is therefore the similarity metric.

2.6 Data Model

Understanding the data model involves understanding the relationship between the images being registered and the statistical noise distribution model, and that relationship affects the shifts between images. According to [7], image registration is a fundamental problem of estimating motion between scenes. This research effort is limited to transformations in translational motion between images of the same scene, i.e., shifts in the x and y directions. In this context [7] notes that motion estimation generally involves observed data that follows

$$R_1(x, y) = I_1(x, y) + N_1(x, y), \tag{2.17}$$

$$R_2(x, y) = I_1(x - \alpha, y - \beta) + N_2(x, y), \tag{2.18}$$

where the measured observation $R_1(x, y)$ is the baseline image consisting of the true image $I(x, y)$ plus its independent noise realization $N_1(x, y)$, $R_2(x, y)$ is the same image shifted $I(x - \alpha, y - \beta)$ plus its independent noise realization $N_2(x, y)$, and α and β represent the shift values in the x and y axis, respectively. In current registration algorithms analyzed, the affect of noise on the observed image is typically modelled as white Gaussian with variance σ^2 [7]. Therein lies a potential fundamental problem in using these types of estimators for generating an image registration algorithm for a laser imaging system where the noise model, or the effect of the noise model, does not follow a Gaussian distribution. The estimation process that determines the shift values α and β typically uses the observed image as the mean of the estimation distribution, thus not accounting for the noise component, or the noise component effect on the true image, that determines what type of distribution is used in the estimation calculation of α and β.

From Equations 2.17 and 2.18 it is evident that the observations R_1 and R_2 have the same mean I, with the only difference being the shifts α and β. Furthermore, in a laser imaging system the actual distributions for N_1 and N_2 are unknown. However, in a LADAR imaging system exhibiting speckle noise, when N_1 and N_2 are added to I, R_1 and R_2 exhibit a negative binomial distribution. As noted before, this is based on Goodman's research of photon count statistics for polarized thermal light and an arbitrary counting interval [4]. Hence, using a better statistical characterization of the effects of speckle noise on the observed image photon count distribution should produce a more accurate registration algorithm for a laser imaging system.

III. Registration Algorithm Design

The development of a registration algorithm optimized for a laser imaging system with speckle noise is fundamentally an estimation problem involving the definition of a probabilistic mapping from the parameter space (the true image scene) to the observation space (the observed image). The problem involves the development of an estimation algorithm that is an inverse mapping from observation space to parameter space which allows the estimation calculation of translational shifts between two images.

This chapter provides a description of the probabilistic mapping and inverse mapping theory behind parameter estimation as indicated in [8], and outlines the design of the registration algorithm using the estimation theory approach.

3.1 Estimation Theory

The new algorithm design begins with deriving the probabilistic mapping between the parameter space of the true image scene and the observation space. This mapping is the foundation of estimation theory, and to gain a basic understanding of this theoretical concept, a model of the estimation problem is proposed that explains the concept of mapping and inverse mapping as indicated in [8, pages 52-54]. Additionally, based on the estimator types described in [8, page 65], a description of the estimator type used in the LITE registration algorithm design is provided.

3.1.1 Parameter Estimation Model. An estimation problem model is graphically represented in Figure 3.1 and has four basic elements: Parameter Space, Probabilistic Mapping, Observation Space, and the Inverse Mapping Estimation Rule [8, page 53].

Parameter Space. The parameter space contains the output variables, or parameters [8]. In the case of the LADAR imaging system, the output variable is the image I generated from the target scene and the associated shifts α and β. However, the true image I is actually an unwanted parameter that is considered a known, non-

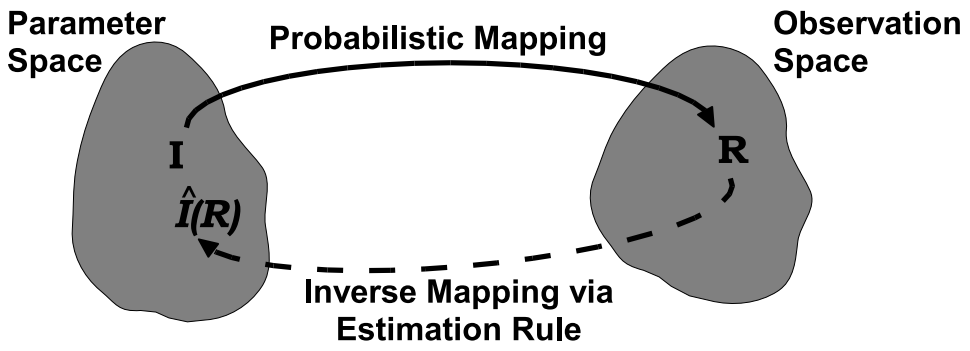


Figure 3.1: **Parameter Estimation Model.** This model illustrates the mapping theory concept behind parameter estimation. If the probabilistic mapping effect on I that creates the observation R is known, then R can be mapped back to create an estimate $\hat{I}(R)$ of I via an estimation rule.

random parameter constant that statistically represents the mean of the photon count distribution observed in R. The coherent nature of the laser beam that creates the speckle noise as described in Chapter II defines the effects on I that are observed in R, which is the Probabilistic Mapping indicated in Figure 3.1.

Probabilistic Mapping. The probabilistic mapping is the "law that governs the effect of I on the observation" [8, page 53]. As previously mentioned in the Problem Statement, speckle noise in a laser imaging system is created by the "coherent nature of the collected energy" [5]. This noise causes the observed image photon count statistics to have a negative binomial distribution. This result is based on Goodman's research of photon count statistics for polarized thermal light and an arbitrary counting interval [4]. From Goodman the distribution is

$$P(R) = \frac{\Gamma(R+M)}{\Gamma(R+1)\Gamma(M)} \left[1 + \frac{M}{\bar{R}}\right]^{-R} \left[1 + \frac{\bar{R}}{M}\right]^{-M}, \tag{3.1}$$

where R is the number of counts observed in a τ-second interval, M "represents the number of 'degrees of freedom' of the distribution intensity included within the measurement interval" [4], and $\bar{R}$ represents the average or mean of the true image. M

can also be thought of as being inversely related to the variance σ^2 of the observed image photon count distribution. The value of M is 50, which is a given parameter determined experimentally using the real images collected from the Laservision system. Hence, Equation 3.1 defines the probabilistic mapping of the output variable I that is the result seen in the Observation Space.

Observation Space. The observation space is simply what is actually observed after the effects of the probabilistic mapping. For LADAR images it is the image collected by the system, denoted as R in Figure 3.1.

Estimation Rule. The goal in estimation theory is to use the observation R to estimate the true value of the variable I from the parameter space, based on what is known about the probabilistic mapping of I. As seen in Figure 3.1, the estimation of I based on the observation R is denoted $\hat{I}(R)$, and the inverse mapping is referred to as the estimation rule [8]. The type of estimator used in the LITE registration algorithm for the estimation rule is discussed next.

3.1.2 Maximum Likelihood Estimation. An abbreviated description of the Maximum Likelihood (ML) estimation procedure described by Van Trees [8, pages 63-65] provides the background for the estimation procedure incorporated in the LITE algorithm design. Van Trees describes the likelihood function as a probability function $p_{r|i}(R|I)$, which is a function of I [8, page 65]. Since the natural logarithm is often used in this estimation procedure, it is sometimes referred to as the log likelihood function [8]. The ML estimator $\hat{I}_{ml}(R)$ is the value of I "at which the likelihood function is a maximum" [8, page 65]. If the maximum value is part of $I's$ parameter space, then the likelihood equation can be determined by differentiating $\ln p_{r|i}(R|I)$ with respect to I and setting equal to zero:

$$\left.\frac{\partial \ln p_{r|i}(R|I)}{\partial I}\right|_{I=\hat{I}_{ml}(R)} = 0. \tag{3.2}$$

This procedure provides the foundational approach for the LITE registration algorithm design.

3.2 Algorithm Design Using Maximum Likelihood Estimation Theory

Since the observations have a negative binomial distribution form as described by Equation 3.1 and the true image I is considered as a non-random parameter constant that statistically represents the mean of the photon count distribution observed in R, then from Equation 3.1, the probability density function (PDF) for the first observed image R_1 is

$$p_{r_1|i}(R_1|I) = \prod_{x=1}^{N}\prod_{y=1}^{N} \frac{\Gamma(R_1(x,y)+M)}{\Gamma(R_1(x,y)+1)\Gamma(M)} \left[1+\frac{M}{I(x,y)}\right]^{-R_1(x,y)} \left[1+\frac{I(x,y)}{M}\right]^{-M}. \tag{3.3}$$

Next the ML estimator $\hat{I}_{ml}$ for image one is derived by first taking the natural logarithm of Equation 3.3, then taking the partial derivative with respect to one discrete point (x_0, y_0):

$$\begin{aligned} \ln\left[p_{r_1|I}(R_1|I)\right] = \sum_{x=1}^{N}\sum_{y=1}^{N} \Bigg[& \ln\left[\Gamma(R_1(x,y)+M)\right] - \ln\left[\Gamma(R_1(x,y)+1)\right] \\ & - \ln\left[\Gamma(M)\right] - R_1(x,y)\ln\left[1+\frac{M}{I(x,y)}\right] \\ & - M\ln\left[1+\frac{I(x,y)}{M}\right] \Bigg] \end{aligned} \tag{3.4}$$

$$\frac{\partial \ln\left[p_{r_1|i}(R_1|I)\right]}{\partial I(x_0,y_0)} = 0-0-0-\frac{\partial}{\partial I(x_0,y_0)}\left[R_1(x,y)\ln\left[1+\frac{M}{I(x,y)}\right]\right] - \frac{\partial}{\partial I(x_0,y_0)}\left[M\ln\left[1+\frac{I(x,y)}{M}\right]\right]$$

$$= -\frac{R_1(x_0,y_0)}{1+\frac{M}{I(x_0,y_0)}}\left[0+\frac{I\cdot 0-M\cdot 1}{I^2}\right]-\frac{M}{1+\frac{I(x_0,y_0)}{M}}\left[0+\frac{M\cdot 1-I\cdot 0}{M^2}\right]$$

$$= \frac{R_1(x_0,y_0)M}{I^2(x_0,y_0)+MI(x_0,y_0)}-\frac{1}{1+\frac{I(x_0,y_0)}{M}}$$

$$= \frac{R_1(x_0,y_0)M}{I^2(x_0,y_0)+MI(x_0,y_0)}-\frac{M}{M+I(x_0,y_0)}, \tag{3.5}$$

which, upon being set equal to zero to solve for the maximum likelihood function, simplifies to

$$0 = \frac{R_1(x_0,y_0)}{1+\frac{M}{I(x_0,y_0)}}\left(\frac{M}{I^2(x_0,y_0)}\right)-\frac{M}{1+\frac{I(x_0,y_0)}{M}}\left(\frac{M}{M^2}\right)$$

$$= \frac{R_1(x_0,y_0)M}{I^2(x_0,y_0)+MI(x_0,y_0)}-\frac{1}{1+\frac{I(x_0,y_0)}{M}}$$

$$= \frac{R_1(x_0,y_0)M}{I(x_0,y_0)\left[I(x_0,y_0)+M\right]}-\frac{M}{I(x_0,y_0)+M}. \tag{3.6}$$

Multiplying the right term by $\frac{I(x_0,y_0)}{I(x_0,y_0)}$, a common denominator is found so that

$$0 = \frac{R_1(x_0, y_0) - MI(x_0, y_0)}{I(x_0, y_0)\left[I(x_0, y_0) + M\right]}. \tag{3.7}$$

Solving for $I(x_0, y_0)$, the ML estimator for a discrete point in image one is simply the observation point of image one

$$\hat{I}_{ml}(x_0, y_0) = R_1(x_0, y_0), \tag{3.8}$$

and it follows that for the entire image the ML estimator is

$$\hat{I}_{ml}(x, y) = R_1(x, y). \tag{3.9}$$

Continuing with image two, Equation 3.9 can be substituted into the PDF for the second image; note that the shift difference between the observed images R_1 and R_2 is accounted for in R_1. Capturing the shifts in this way allows for the isolation of α and β in order to solve for their respective estimators.

$$p_{r_2|\hat{i},\alpha,\beta}(R_2|\hat{I}_{ml},\alpha,\beta) = \prod_{x=1}^{N}\prod_{y=1}^{N}\left[\frac{\Gamma(R_2(x,y)+M)}{\Gamma(R_2(x,y)+1)\Gamma(M)}\left[1+\frac{M}{\hat{I}_{ml}(x-\alpha,y-\beta)}\right]^{-R_2(x,y)}\left[1+\frac{\hat{I}_{ml}(x-\alpha,y-\beta)}{M}\right]^{-M}\right]$$

$$= \prod_{x=1}^{N}\prod_{y=1}^{N}\left[\frac{\Gamma(R_2(x,y)+M)}{\Gamma(R_2(x,y)+1)\Gamma(M)}\left[1+\frac{M}{R_1(x-\alpha,y-\beta)}\right]^{-R_2(x,y)}\left[1+\frac{R_1(x-\alpha,y-\beta)}{M}\right]^{-M}\right]. \quad (3.10)$$

Next, to derive ML estimators for α and β, a log-likelihood is calculated by taking the natural logarithm of Equation 3.10,

$$\ln[p_{r_2|i,\alpha,\beta}(R_2|R_1,\alpha,\beta)] = \sum_{x=1}^{N}\sum_{y=1}^{N}\Bigg[\ln\left[\Gamma(R_2(x,y)+M)\right] - \ln\left[\Gamma(R_2(x,y)+1)\right] - \ln\left[\Gamma(M)\right] - R_2(x,y)\ln\left[1+\frac{M}{R_1(x-\alpha,y-\beta)}\right] - M\ln\left[1+\frac{R_1(x-\alpha,y-\beta)}{M}\right]\Bigg]. \quad (3.11)$$

By inspection of the terms in Equation 3.11, it is clear that the first three are not affected by α and β. Thus, they will not affect the value of the log-likelihood summation. The fifth term is affected by the α and β shifts only around the edges of the image. At the edges pixel values could potentially vary significantly depending on the intensity variations between images along the edges. This could potentially add or subtract significantly in the summation. However, because the Laservision sys-

tem typically captures images well within the laser beam's FOV leaving fairly dark borders, it is assumed that the edge effects are negligible and the fifth term can be ignored. Hence, Equation 3.11 simplifies to a correlation term which results in

$$L(\alpha,\beta) = \sum_{x=1}^{N}\sum_{y=1}^{N}\left[-R_2(x,y)\ln\left[1+\frac{M}{R_1(x-\alpha,y-\beta)}\right]\right]. \tag{3.12}$$

Then by combining the terms of the natural log argument over a common denominator and taking the inverse, the negative sign is eliminated via the log rule $\ln(\frac{1}{x}) = -\ln(x)$ to produce

$$L(\alpha,\beta) = \sum_{x=1}^{N}\sum_{y=1}^{N}\left[R_2(x,y)\ln\left[\frac{R_1(x-\alpha,y-\beta)}{R_1(x-\alpha,y-\beta)+M}\right]\right]. \tag{3.13}$$

Simply put, the α and β shifts that create a peak value of the correlation function are the estimators for α and β:

$$(\hat{\alpha},\hat{\beta}) = \underset{(\alpha,\beta)}{argmax}\left[L(\alpha,\beta)\right]. \tag{3.14}$$

This new algorithm design is called the Laser-illuminated Translation Estimation (LITE) registration algorithm.

Since the LITE algorithm is tested against the cross-correlation algorithm, a quick comparative examination of the two algorithm designs is helpful. First, the LITE algorithm can be simply manipulated by allowing the log function term to equal:

$$G(x-\alpha,y-\beta) = \ln\left[\frac{R_1(x-\alpha,y-\beta)}{R_1(x-\alpha,y-\beta)+M}\right], \tag{3.15}$$

which could be performed as a pre-processing step as part of the actual computer implementation of the algorithm. This adjustment allows the LITE algorithm to be expressed as

$$L(\alpha, \beta) = \sum_{x=1}^{N} \sum_{y=1}^{N} \left[R_2(x, y) G(x - \alpha, y - \beta) \right], \tag{3.16}$$

which can now be compared to the cross-correlation algorithm from Appendix A

$$L(\alpha, \beta) = \frac{1}{\sigma_n^2} \sum_{x=1}^{N} \sum_{y=1}^{N} \left[R_2(x, y) R_1(x - \alpha, y - \beta) \right]. \tag{3.17}$$

This brief comparison simply highlights the fact that the LITE algorithm is procedurally similar to the cross-correlation, with the exception of the added division and natural logarithm floating point operations. Unfortunately, while this should not add significantly to overall processing time, it is not likely to provide a faster algorithm design for the Laservision system. However, if the performance improvement over the cross-correlation algorithm is significant enough, it will demonstrate that implementing true statistics in the registration algorithm design improves overall laser image registration accuracy. This result ultimately serves to improve speckle noise suppression to produce a single image with adequate resolution to allow detection and identification algorithms to operate accurately or to allow system operators to perform target detection and identification within a scene.

IV. Performance Results and Analysis

According to Robinson [7], performance characterization is critical to measuring algorithm optimization. For the comparative tests a pixel standard deviation error measurement metric on the algorithm itself versus another registration algorithm is used, as well as consideration of the processing time performance metric analyzed by [5].

Performance verification of the LITE algorithm is conducted on test images with simulated speckle noise versus one of the top two algorithms analyzed by [5], the cross-correlation algorithm. The performance metric used for the simulated testing is registration standard deviation error measured in pixels over varying signal-to-noise ratios (SNRs) (e.g. light intensity levels). Further test analysis comparing the two algorithms is conducted using real-world ground test images containing the actual speckle noise effects of the Laservision imaging system. The performance metric used for the real data testing is registration standard deviation error metric measured as light intensity at each pixel over varying SNRs.

This chapter outlines the testing methodology incorporated for both the simulated speckle noise images and the real-world images with speckle, and the results and analysis for both the simulated data and the real data.

4.1 Testing Methodology

4.1.1 Simulated Speckle Noise Image Testing. The simulated speckle noise image data is developed to perform a comparative analysis of the LITE algorithm versus a cross-correlation algorithm. The comparative analysis evaluates the improvement realized by implementing the negative binomial distribution in the parameter estimation versus the traditional Gaussian distribution assumption. Using the MATLAB® negative binomial random generator function, two independent negative binomial noise realizations were added to the true image and a copy of the true image. Figures 4.1 and 4.2 illustrate the twin images before and after the addition of the simulated speckle noise. The image in Figure 4.1 is generated using the real-world images

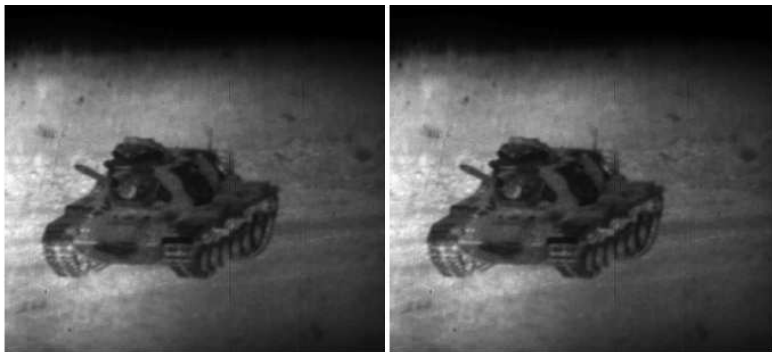


Figure 4.1: **True Image Scene without Noise.** This represents the unshifted pair of noiseless images used for the simulated speckle noise testing and analysis.

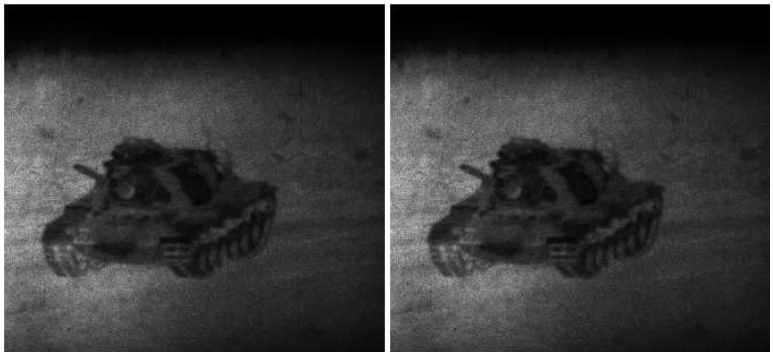


Figure 4.2: **Twin Images with Speckle Noise.** This represents the unshifted pair of images after the simulated speckle noise is added. Each image has a random, independent realization of the speckle noise.

collected by Laservision, which are processed to provide a noiseless image that is used to create the simulated speckle noise images in Figure 4.2. Simulating the negative binomial noise in a noiseless scene serves to test the algorithms against strictly speckle noise, and it avoids the potential introduction of additional noise sources that may be present in a real-world image. This simulation environment tests the LITE algorithm, designed only with speckle noise statistics, and the cross-correlation algorithm, designed with Gaussian noise statistics.

Since the images are copies of each other, there are no translational shifts between them; i.e., they are in effect registered. This condition requires the registration algorithms to correct only for false shifts created by the speckle noise, thus isolating

the algorithm performance comparison to speckle noise generated pixel shift errors. Also, the degree of freedom value M influences the degree to which the speckle noise affects the image, and the value used is fifty, which is calculated via experimentation as indicated earlier. Furthermore, the average SNR value for this scene is measured as approximately six, which establishes the upper end of the SNR range used for the comparison.

The two images with independent speckle noise realizations are registered using a standard cross-correlation algorithm and the LITE algorithm, both implemented in MATLAB®. The procedure of adding independent noise realizations to each image and registering with the two algorithms is repeated one-thousand times to simulate one-thousand independent noise realizations and registrations at each SNR value. The registration shifts estimated by each algorithm for each noise realization are used to calculate a registration standard deviation error metric of the x and y shifts calculated by each algorithm, measured in integer pixel values. The noise realization and registration steps above are repeated for each SNR value to produce a comparison of the algorithms by calculating the pixel shift standard deviation for each algorithm:

$$\sigma_{reg} = \sqrt{\frac{\sum_{n=1}^{N} [dx_n^2 + dy_n^2]}{N}}, \tag{4.1}$$

where dx and dy represent the measured shift values in the x and y directions, respectively, and N is the number of the noise realizations calculated.

The light intensity signal-to-noise ratio (SNR) of the images is varied from 0.8 to 7 to simulate very low to very high light levels. At the upper end of this range it is anticipated that the algorithms will perform equally well. This expectation is due to the higher light intensity level, which causes the negative binomial statistical distribution of the speckle noise to begin resembling a Gaussian distribution. Thus, most registration algorithms will perform equally well, particularly the cross-correlation

algorithm, which is known to be the best estimator for signals with Gaussian noise (ref Appendix A). Similarly, for intensity levels that are significantly lower, the noise is significant enough that it is anticipated the algorithms will perform equally poorly. Varying the SNR light intensity levels provides a means to analyze the algorithms and determine if there is an optimum SNR range where the LITE registration algorithm performs better than the cross-correlation algorithm.

4.1.2 Real-world Speckle Noise Image Testing. The comparative analysis of the LITE algorithm versus a cross-correlation algorithm is continued on a set of the real-world data provided from the Laservision ground tests. The objective of testing and analyzing the two algorithms with real data is to determine if the LITE algorithm outperforms the cross-correlation algorithm within the range that the simulated data testing indicates (i.e., between approximately 3 - 5.5 average SNR) for real-world images. Figure 4.3 illustrates the target scene of a truck with a resolution board that is used in this testing.

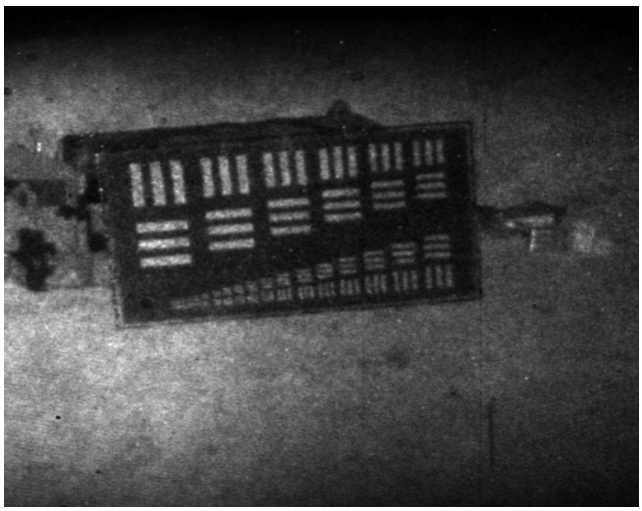


Figure 4.3: **Real-world Target Scene.** This scene is the real-world ground test data collected by the Laservision system of a truck with resolution target board, and is the scene used for the LITE vs. Cross-correlation comparison and analysis.

Since the LITE algorithm model is designed with and the simulated data is created with only speckle noise statistics, this scene represents the shortest range (3 km) that the Laservision system was tested that should contain mostly, if not only, speckle noise. In other words, this data set should minimize any other noise sources (i.e., atmospheric turbulence, thermal, etc.) that could cause a performance comparison anomaly. Also, the resolution board provides sub-regions which are used to find lower average SNR scenes within the desired average SNR range.

Fifty real-world images of the truck with resolution target board are used from the Laservision image data base; each contains random, independent speckle noise realizations and unknown shifts between the images. Since the true translational shifts are not known, the registration shifts estimated by each algorithm for each noise realization are used to calculate a registration standard deviation error metric measured as light intensity at each pixel. Thus, the error metric calculation for each algorithm is

$$\sigma_i = \sqrt{\frac{\sum_{k=1}^{K}\sum_{x=1}^{M}\sum_{y=1}^{N}[D_k(x+\alpha_k, y+\beta_k) - I(x,y)]^2}{KMN}}, \tag{4.2}$$

where D_k is each image with its associated shift value and I represents the mean of the images intensity level at each pixel. This mean is determined when the algorithm registers all the images and they are averaged, thus creating one set of light intensity mean values for each pixel location averaged for all fifty images. K is the number of images used for registration (fifty), and M and N are the dimensions of the image in the x and y directions, i.e., the number of pixels per side, which are 500 x 500 for the full scene and 42 x 42 for the smaller scenes. The reason for using this smaller image scene is explained later.

The first real image tested is the full scene as shown in Figure 4.3, which has a measured average SNR of 6.02. This image provides a sample point at the upper

bound of the average SNR range. As in the simulated data, the results are anticipated to show both algorithms will perform equally well, mainly because the speckle noise begins to resemble a Gaussian distribution at higher light intensity levels as described earlier. The goal then is to find average SNRs within this image scene that falls within the window where the LITE algorithm is expected to show improvement, which is where the smaller image scenes come into use for the testing scenario.

The full scene is scanned to find smaller regions with lower average SNRs to find representative samples across the desired SNR range (3 - 5.5). Furthermore, the sample scenes were chosen such that the resolution bars were as close to the center as possible leaving as dark of border as possible. This selection criteria was used to minimize the border effects that were assumed to be minimal in Chapter III, thus allowing both algorithms to perform as optimally as possible within their designs. An exhaustive search is performed using the selection criteria above and four sample scenes, each 42 x 42 pixels, are collected with average SNRs of 4.17, 4.36, 4.44, and 5.15. Note that 4.17 was the lowest average SNR found that met the selection criteria. However, the samples at 4.36 and 4.44 are in the heart of the testing window where the simulated data showed a peak improvement of approximately 0.16 pixel around 4.4 average SNR. This average SNR is where the LITE algorithm is expected to have the best performance improvement over the cross-correlation algorithm.

4.2 Results and Analysis

4.2.1 Simulated Speckle Noise Data. Figure 4.4 illustrates the registration error of the LITE registration algorithm versus that of the cross-correlation algorithm across the varying light intensity SNRs. As anticipated, the upper range (approximately 5.5 and greater) clearly demonstrates that the two algorithms begin to perform equally as well with little error in the shift estimations. Also clearly evident is that for very low SNRs (approximately below 2) the two algorithms are performing equally poorly. In other words, the noise is significant enough that the pixel error is significant for both algorithms, with little to no difference between them. However, within

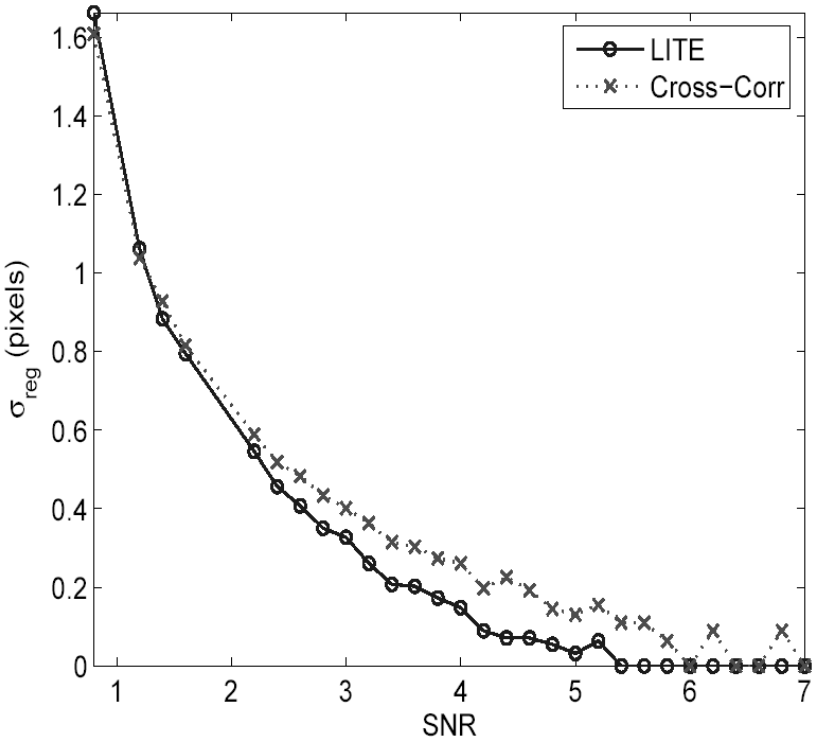


Figure 4.4: **Simulated Data Registration Error Standard Deviation, LITE vs. Cross-Correlation.** The simulated data performance results demonstrate that the LITE algorithm shows improved performance over cross-correlation, specifically within the 3 to 5.5 SNR range, where there is an average improvement of 0.1 pixels with a peak of approximately 0.16 around 4.4 average SNR.

the SNR range of 3 to 5.5 the LITE algorithm has improved performance over the cross-correlation algorithm. The average improvement in this range is 0.1 pixels, with a peak of approximately 0.16 around 4.4 average SNR.

4.2.2 Real-world Speckle Noise Data. Figure 4.5 shows the real-data full-scene of the truck and resolution target board, and a side-by-side comparison of the LITE versus Cross-correlation algorithms both visually and with the light intensity standard deviations noted above each image. This comparison illustrates that there is little to no difference visually between the two algorithms at the higher average SNR range, as expected. Also, the light intensity standard deviations are comparable at this average SNR. Unexpectedly, though, at the other average SNR values of 4.17,

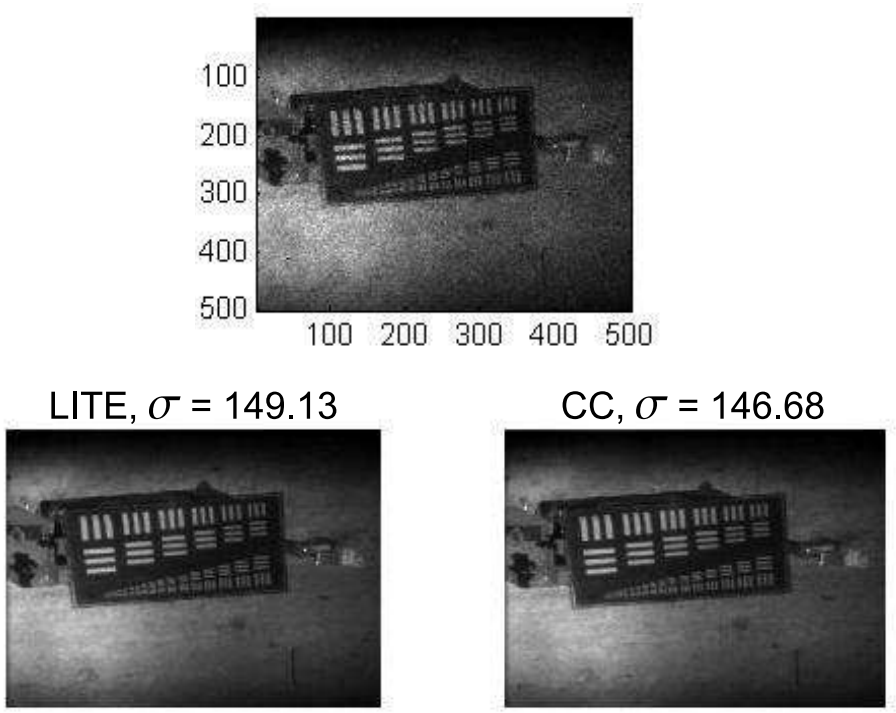


Figure 4.5: **LITE vs. Cross-correlation Comparison, Full Scene, Average SNR = 6.** This side-by-side comparison illustrates that there is little to no difference visually between the two algorithms at the higher average SNR range, as expected. The light intensity standard deviations are also comparable at this average SNR.

4.36, 4.44 and 5.15, the two algorithms are performing nearly identically as well, with the Cross-correlation showing a very negligible advantage. Figures 4.6, 4.7, 4.8, and 4.9 highlight this unfortunate outcome.

Finally, Figure 4.10 depicts pixel intensity standard deviations versus average SNR for each algorithm to graphically highlight the unexpected results for the real-data testing. There are potentially two initial reasons that may have caused this outcome, which are discussed in the conclusions of Chapter V.

4.2.3 Processing Time. As mentioned earlier, the processing time performance metric analyzed by [5] is considered, which is measured by implementing the

basic Matlab® tic and toc functions around the main loops for each algorithm. Surprisingly, the LITE algorithm appears to consistently run slightly faster than the Cross-correlation algorithm. Specifically, for processing the larger 500 x 500 images, the LITE algorithm performed approximately 13% faster, and for the smaller 42 x 42 scenes it performed approximately 17% faster. The potential reasons for this result are discussed further in the conclusions of Chapter V.

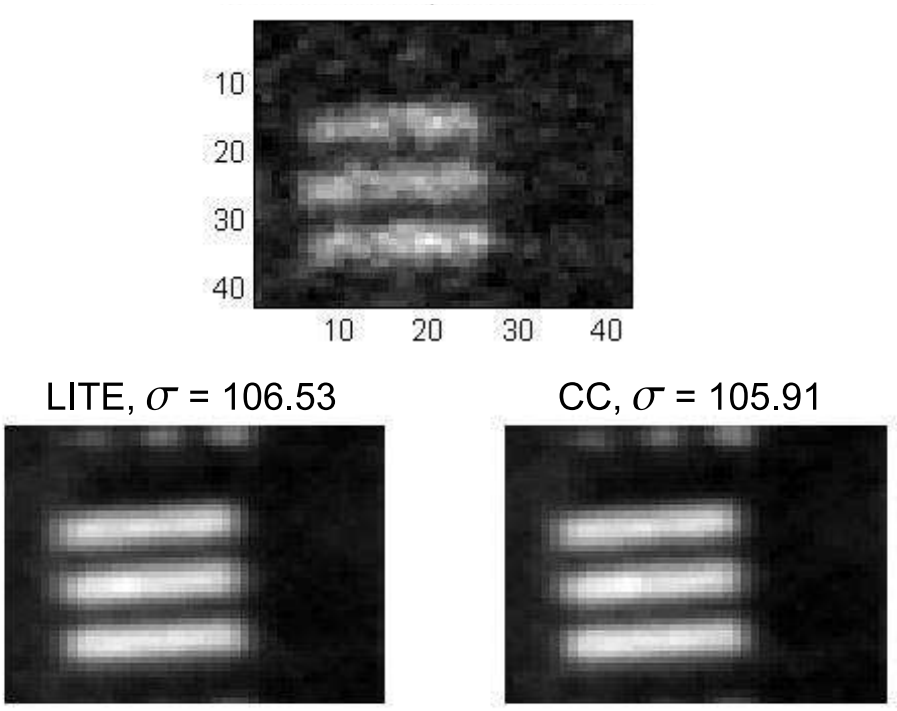


Figure 4.6: **LITE vs. Cross-correlation Comparison, Sub-Region Scene, Average SNR = 4.17.** This comparison illustrates that there is an unexpected result showing little to no difference either visually or between the light intensity standard deviations at this lower average SNR.

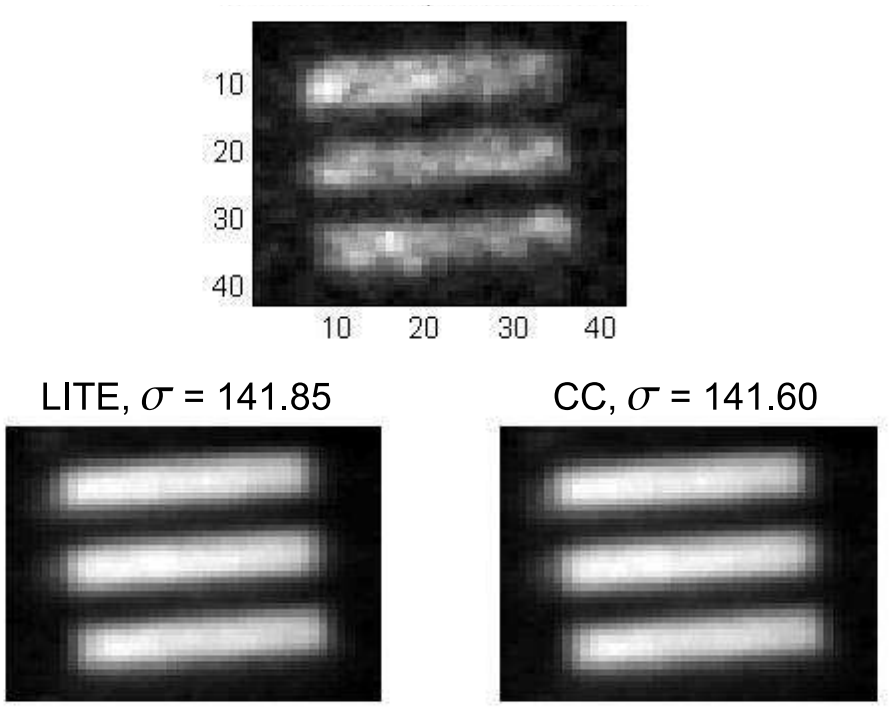


Figure 4.7: **LITE vs. Cross-correlation Comparison, Sub-Region Scene, Average SNR = 4.36.** This comparison illustrates that there is an unexpected result showing little to no difference either visually or between the light intensity standard deviations at this average SNR that is near the value that the simulated data showed where the LITE with the most improvement.

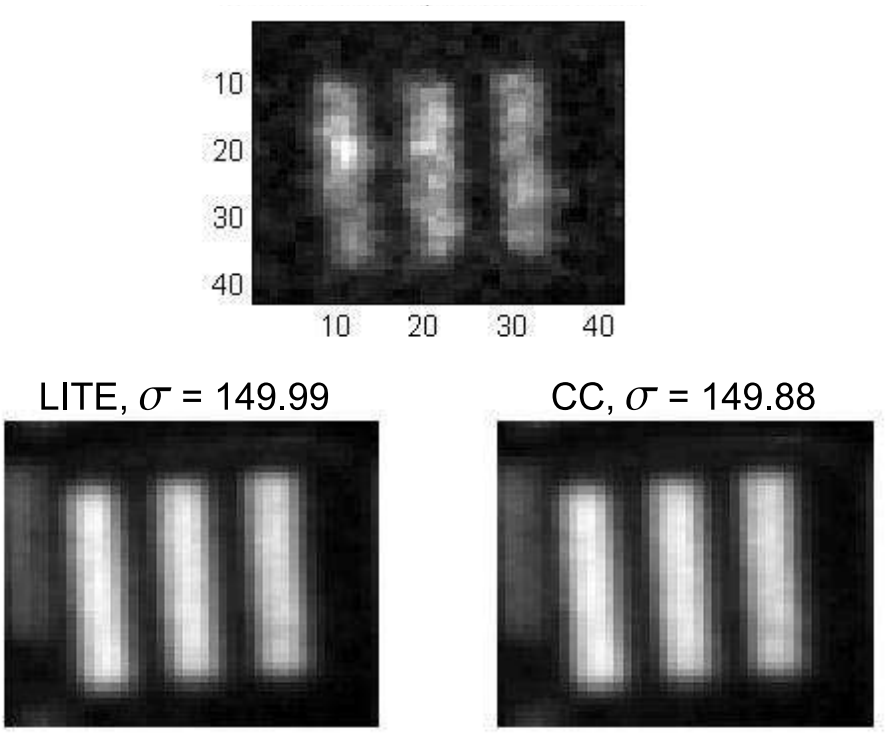


Figure 4.8: **LITE vs. Cross-correlation Comparison, Sub-Region Scene, Average SNR = 4.44.** This comparison illustrates that there is an unexpected result showing little to no difference either visually or between the light intensity standard deviations at this average SNR that is at the value that the simulated data showed where the LITE with the most improvement.

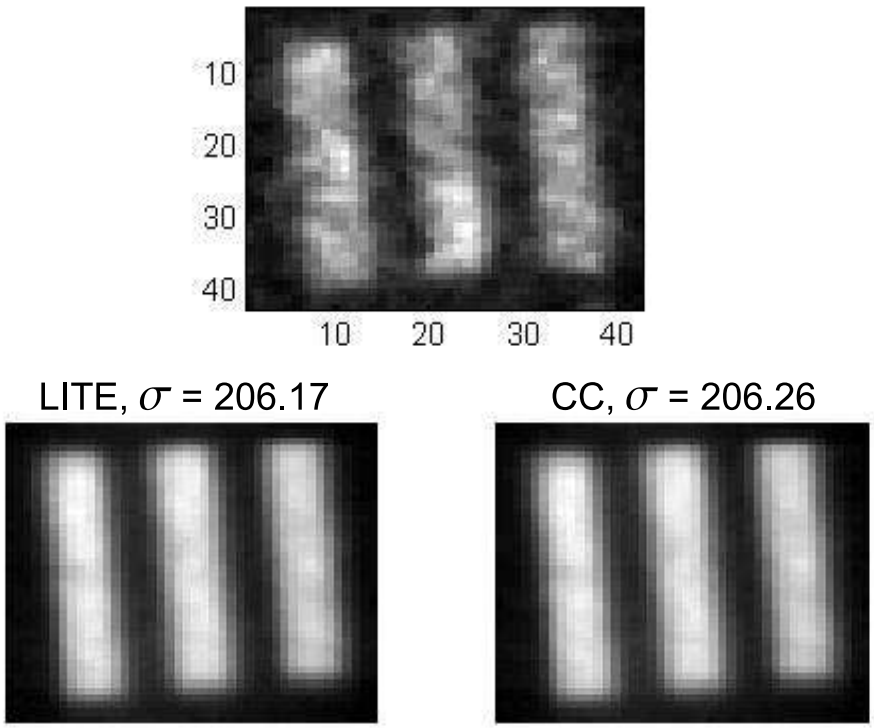


Figure 4.9: **LITE vs. Cross-correlation Comparison, Sub-Region Scene, Average SNR = 5.15.** This comparison illustrates that there is an unexpected result showing little to no difference either visually or between the light intensity standard deviations at this average SNR that is near the value that the simulated data showed where the LITE with the most improvement.

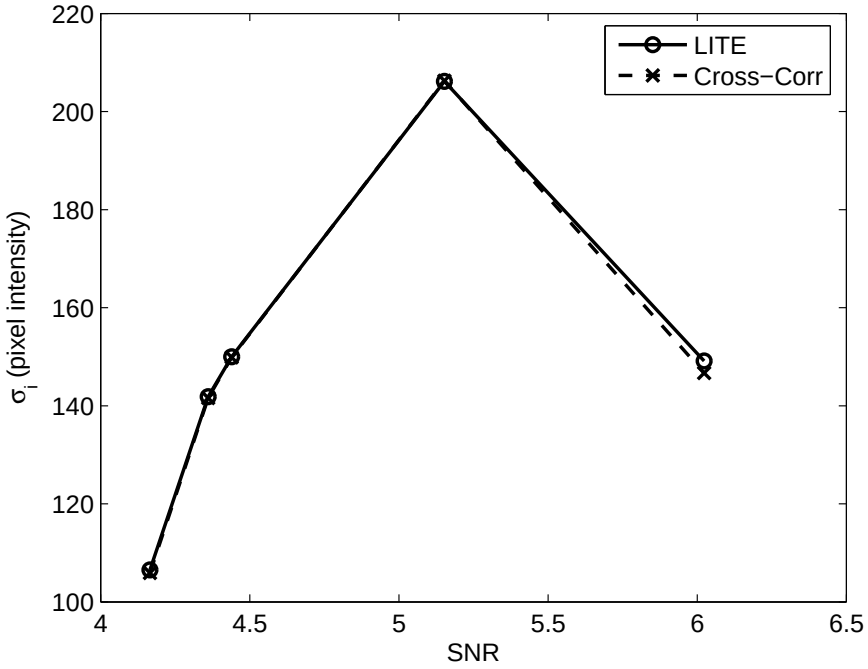


Figure 4.10: **Real-world Data Pixel Intensity Standard Deviation, LITE vs. Cross-Correlation.** The real-world data performance results demonstrate that the LITE algorithm did not outperform the Cross-correlation algorithm as expected, but rather did as well in the heart of the average SNR range, and negligibly worse at the upper average SNR range. The two potential reasons are discussed in the conclusions of Chapter V.

V. Conclusions

This research effort focuses on developing a robust optical image registration algorithm for negating speckle noise effects in coherent images generated by a laser imaging system, named the Laser-illuminated Translation Estimation (LITE) registration algorithm. This chapter provides closing remarks beginning with a recap of the highlights of the research and conclusions about the LITE algorithm, and finishing with potential areas for future research.

5.1 Summary

The concept of image registration tailored for coherent laser imaging systems is very new, and research on related registration algorithms includes a brief review of the top two image registration algorithms analyzed by [5], the cross-correlation and vector-block methods. The cross-correlation method is "generally regarded as a robust estimator in the presence of noise" [2], but the drawback is that the use of FFTs requires more computer resources than the Laservision system can allocate, and it is known that cross-correlation is the best estimator for Gaussian noise (ref Appendix A). The vector-block method does reduce the cross-correlation computational load due to reducing the dimensionality of the calculations from N^2 to 2 x N; however, the fundamental problem of the assumed Gaussian speckle noise distribution model still exists for this algorithm. The striking results from the comparison study of these two algorithms (ref Appendix B) is that the processing speed of the vector-block method is over two times faster due to the dimensionality reduction of the calculations. Equally striking is the extremely small difference in MSE performance, which translates into maintaining the quality in the final image product.

The system description highlighted the AFRL Laservision system, which illuminates a scene with a laser beam, then receives the reflected laser energy using an optical telescope which feeds a CCD detector. The receiver is designed to capture ten images per second of the target scene, and it is these ten essentially identical images, offset by only translational motion, that must be properly registered. Also,

the coherent nature of laser light causes the speckle noise phenomenon, and no known algorithm in the past has been specifically designed to register laser generated images using negative binomial statistics in the shift estimation algorithm design.

Image registration is typically defined as a process that compares two or more images by calculating differences in translation (horizontal and vertical shifts), rotation, and scaling in order to match or align the images for further processing [6], and it is a key initial image processing step before target detection techniques can be implemented. Also, recall that according to Robinson [7], image registration is a fundamental problem of estimating motion between scenes, and the algorithm design is fundamentally an estimation problem involving a probabilistic mapping from the parameter space (the true image scene) to the observation space (the observed image). This problem involves the development of an algorithm that is an inverse mapping from observation space to a parameter space that allows the calculation of the translational shifts between two images.

5.2 Conclusions

The LITE registration algorithm is designed using speckle noise statistics. It is found via simulation that using the correct statistics produces a registration algorithm with improved performance relative to an algorithm derived using noise distributions that are not consistent with the data they are processing; i.e., the white Gaussian noise assumption. However, the real data comparison tests did not provide the results verifying that using an algorithm designed specifically with the true speckle noise statistics outperforms an algorithm design not characterized with the true noise statistics. A couple of hypothesized explanations are offered in the real data conclusions.

5.2.1 Simulated Data. The simulated data results indicate that the LITE algorithm shows promise of being valuable for low light level imaging scenarios, specifically in the 3 to 5.5 SNR range. The average improvement in this SNR range is 0.1

pixels, with a peak of approximately 0.16 around 4.5 SNR. This result may not appear to be very significant; however, at longer ranges it could prove valuable in suppressing speckle noise enough to be noticeable to either the human eye for operator target identification or to allow improved target detection and recognition algorithms to perform better.

Further observations indicate that at very low light levels (i.e., below 2 SNR) the noise is significant enough that both algorithms perform equally as poorly in that the pixel error standard deviation becomes progressively worse for both, with little to no difference between them. Likewise, at high light levels the two algorithms perform equally well since the probability density function of the data becomes similar to a Gaussian distribution. In these situations traditional registration algorithms behave similarly to the LITE algorithm and might be preferable in that they are faster and require fewer floating point operations to execute. However, the research conducted to develop the LITE registration algorithm demonstrates significant promise in implementing true statistics in the registration algorithm design, thus improving overall laser image registration, and it has clearly laid the groundwork for continued research.

5.2.2 Real Data. The real data results indicate that the LITE algorithm did not outperform the Cross-correlation algorithm as expected, but rather did as well in the center of the average SNR range, and negligibly worse at the upper average SNR range. The two potential reasons for this outcome are either there are other noise sources within the real data beyond only speckle, or the calculation of the M value provided for the testing is not accurate. If there are other noise sources present in the data set, then this will cause the LITE algorithm to perform less effectively because it is designed specifically with speckle noise statistics and assumes that the short-range real data set contained only, or mostly, negative binomial speckle noise. The M value was extrapolated from the Laservision image data base set because the system was not instrumented during the ground tests to measure and characterize the sensor with respect to its M value. A quick test and analysis varying the M

value in the Matlab® code from 1 to 100 provided no significant changes in the LITE algorithms performance. If the M value is actually much higher than calculated, then this will have the same effect as increasing the SNR light intensity level, which in turn causes the statistical distribution of the noise to begin resembling Gaussian noise. Thus, the Cross-correlation algorithm would tend to outperform the LITE algorithm in this situation. Further analysis to determine the true M value may be necessary, as well as an in depth analysis of the data sets to quantify what other noise sources may be present. The Future Research section discusses some ideas on accounting for other noise sources and possibly creating an adaptive, more dynamic algorithm by incorporating the M value into the estimation process.

5.2.3 Processing Time. The processing time performance metric analyzed by [5] is measured by implementing the basic Matlab® tic and toc functions around the main loops for each algorithm. Initial indications are that the LITE algorithm appears to consistently run slightly faster than the Cross-correlation algorithm. Specifically, for processing the larger 500 x 500 images, the LITE algorithm performed approximately 13% faster, and for the smaller 42 x 42 scenes it performed approximately 17% faster. This result was not expected, but upon further analysis of the Matlab® code, the potential reasons for this may lie in the number of operations that are accomplished outside the main shift estimation algorithm loop for each of the registration algorithms. For the LITE algorithm there are four operations involving the logarithm, conjugate, and FFT that are one time operations performed outside of the loop. Specifically, there is only one FFT operation executed and it is accomplished outside the loop. Conversely, the Cross-correlation algorithm has one FFT outside the loop and another inside the loop along with a conjugate operation.

5.3 Avenues for Future Research

There are several areas that are ripe for continuing with this particular algorithm design or for branching off into similar fields. The following is a brief list of the avenues

available for future research, along with a short description of what research could be conducted in each area.

5.3.1 Cramer-Rao Lower Bound Analysis. Probably the most obvious future sub-research area that could benefit laser image registration algorithm design involves calculation of the Cramer-Rao Lower Bound. According to [7], the Cramer-Rao bound helps to characterize "the 'difficulty' with which a set of parameters can be estimated using a given data model from an information theoretic standpoint." Therefore, the calculation would provide a lower bound to how well a laser image registration algorithm should be expected to perform and could determine if any more effort should be expended on continuing to improve the LITE algorithm.

5.3.2 Adaptive Joint Estimator Design. A very promising avenue is to redesign the estimator to adaptively account for both varying values of M and the possibility of prior knowledge of I. An adaptive joint estimator incorporating M, I, α, and β would significantly boost the robustness of the LITE registration algorithm.

5.3.3 Other Transformation Effects. Another obvious future route is analyzing the effects of scaling and rotation transformations between images. Depending on the environment in which a laser imaging system is operated and on the platform on which it is installed, significant gains could be made by incorporating rotation and scaling compensation into the registration shift estimation algorithm design.

5.3.4 Long-range Scintillation Noise Effects. Yet another obvious avenue is examining the atmospheric turbulence effects on the long-range data that create additional scintillation noise on top of the speckle noise. In this area LITE algorithm design robustness could be significantly advanced by incorporating the turbulence statistics in the estimation process.

5.3.5 Pre-Processing Technique Analysis. As part of improving the overall computational speed of the algorithm, there may be analog pre-processing techniques

which could be implemented to accelerate the calculations of the LITE algorithm. Specifically, an analog pre-filter designed to perform the log function of the LITE algorithm could be implemented in the image processing chain before the analog signal passes into the A/D converters.

5.3.6 Beam, Platform, and Target Motion Effects. Another ripe area of research within laser image processing is to develop an algorithm or redesign the LITE algorithm to distinguish between beam, platform, and target motion that may be present in a laser imaging system. The algorithm would need to quantify each motion independently to allow for independent compensation techniques that would prevent the registration algorithm from falsely locking onto any of these false motion effects.

Appendix A. Shift Parameter Estimation Using Gaussian Noise

This appendix is provided as additional background both for the cross-correlation registration algorithm and estimation theory. As part of a background self-study, an analysis was conducted using a Gaussian noise assumption in the shift parameter estimation theory as described in Chapter III.

The following are the analytical calculations and their implications for this research; they demonstrate why the cross-correlation is the best shift estimator for a signal with white Gaussian noise.

> ! NOTE: In this thesis, the development of the new registration algorithm is limited in that it considers only translational shifts (i.e., only those shifts in the vertical and horizontal directions). Hence, the shift estimation conducted in this appendix is also limited to translational shifts.

A.1 Estimation Theory Using the Gaussian Noise Assumption

In the Data Model section in Chapter II, the observed images R_1 and R_2 are expressed as

$$R_1(x, y) = I(x, y) + N_1(x, y), \tag{A.1}$$

$$R_2(x, y) = I(x - \alpha, y - \beta) + N_2(x, y). \tag{A.2}$$

where the observations of the Rs are the light intensity or photon counts in each pixel of the measure target scene.

The Parameter Estimation Model for this analysis assumes that the noise components in Equations A.1 and A.2 are white Gaussian; the noise is distributed as a normal distribution with zero mean and variance $N_i(x, y) \sim N(0, \sigma^2)$, thus N_1 and N_2 are

$$N_i = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[\frac{-(R_i(x, y) - I(x, y))^2}{2\sigma^2}\right]. \tag{A.3}$$

From this assumption Equation A.3 is used in the probabilistic mapping between the parameter space and the observation, and the PDFs of R_1 and R_2 are

$$p_{r_1|i}(R_1|I) = \prod_{x_1=1}^{N} \prod_{y_1=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} e^{\left[\frac{-(R_1(x_1,y_1)-I(x_1,y_1))^2}{2\sigma^2}\right]} \tag{A.4}$$

$$p_{r_2|i,\alpha,\beta}(R_2|I,\alpha,\beta) = \prod_{x_2=1}^{N} \prod_{y_2=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} e^{\left[\frac{-(R_2(x_2,y_2)-I(x_2-\alpha,y_2-\beta))^2}{2\sigma^2}\right]}. \tag{A.5}$$

From Chapter III, the goal in estimation theory is to use the observation R to estimate the true value of the variable I from the parameter space, based on what is known about the probabilistic mapping of I. The inverse mapping from observation space back to the parameter space is accomplished via the estimation rule [8], and various estimators are used for this rule. However, for this shift estimation problem the maximum likelihood estimator is selected, and is discussed next.

A.2 Maximum Likelihood Estimation

Before delving into the derivation of the algorithm, a short description of Maximum Likelihood (ML) estimation is provided. The ML estimator is used as an estimation rule for known real, non-random parameter estimation and is used as one type of estimation procedure that measures how well the estimation performs. Van Trees describes the likelihood function as a probability function $p_{r|i}(R|I)$, which is a function of I [8, page 65]. Since the natural logarithm is often used in this estimation procedure, it is sometimes referred to as the log-likelihood function [8]. The ML estimator $\hat{I}_{ml}(R)$ is defined as the value of I "at which the likelihood function is a maximum" [8, page 65]. If the maximum value is part of $I's$ parameter space, then the likelihood equation can be determined by differentiating $\ln p_{r|i}(R|I)$ with respect to I and setting the result equal to zero.

A.3 Algorithm Design Using the ML Estimator and Gaussian Noise Assumption

The design for the shift parameter estimation begins with implementing the Probabilistic Mapping from above using the Gaussian noise distribution assumption from Equations A.4 and A.5. Then a maximum likelihood (ML) estimator is incorporated in deriving the log-likelihood estimators for the α and β shifts. The design is based on using two images: a baseline true image and a secondary, shifted true image as described by R_1 and R_2 in Equations A.1 and A.2.

Consider first the observation R_1 of first image containing no shift and the assumed Gaussian noise. Then, from Equations A.4, the PDF for the entire image is

$$p_{r_1(x,y)|i_1(x,y)}(R_1(x,y)|I_1(x,y)) = \prod_{x=1}^{N}\prod_{y=1}^{N}\frac{1}{\sqrt{2\pi}\sigma_n}e^{\left[\frac{-(R_1(x,y)-I_1(x,y))^2}{2\sigma_n^2}\right]} \tag{A.6}$$

Next a ML estimator I_{ml} is derived for I_1 by first taking the natural logarithm of Equation A.6, then taking the derivative for one discrete point (x_0, y_0) and setting the result equal to zero:

$$\ln\left[p_{r_1|i_1}(R_1|I_{ml})\right] = \sum_{x=1}^{N}\sum_{y=1}^{N}\left[\ln\left(\frac{1}{\sqrt{2\pi}\sigma_n}\right) - \frac{(R_1(x,y)-I_{ml}(x,y))^2}{2\sigma_n^2}\right] \tag{A.7}$$

$$\frac{\partial \ln\left[p_{r_1|i_1}(R_1|I_1)\right]}{\partial I_{ml}(x_0,y_0)} = -\frac{R_1(x,y)-I_{ml}(x,y)}{\sigma_n^2} = 0. \tag{A.8}$$

Solving for $I(x_0, y_0)$, the ML estimator for a discrete point in image one is simply the observation point of image one, or

$$\hat{I}_{ml}(x_0,y_0) = R_1(x_0,y_0), \tag{A.9}$$

and it follows that for the entire image the ML estimator is

$$\hat{I}_{ml}(x,y) = R_1(x,y). \tag{A.10}$$

Continuing with image two, Equation A.10 is substituted into Equation A.2 to yield

$$R_2(x,y) = \hat{I}_{ml}(x-\alpha, y-\beta) + N_2(x,y) \tag{A.11}$$

Next, using Equation A.11 in the PDF of Equation A.5 for the second image, and substituting in Equation A.10 results in

$$\begin{aligned} p_{r_2|\alpha,\beta}(R_2|\alpha,\beta) &= \prod_{x=1}^{N}\prod_{y=1}^{N} \frac{1}{\sqrt{2\pi}\sigma_n} e^{\left[\frac{-(R_2(x,y)-I_{ml}(x-\alpha,y-\beta))^2}{2\sigma_n^2}\right]} \\ &= \prod_{x=1}^{N}\prod_{y=1}^{N} \frac{1}{\sqrt{2\pi}\sigma_n} e^{\left[\frac{-(R_2(x,y)-R_1(x-\alpha,y-\beta))^2}{2\sigma_n^2}\right]}. \end{aligned} \tag{A.12}$$

Note that the shift difference between the observed images R_1 and R_2 is accounted for in R_1, which allows for the isolation of α and β in order to solve for their respective estimators.

Next, to derive ML estimators for α and β a log-likelihood is calculated by taking the natural logarithm of Equation A.12,

$$\begin{aligned} \ln\left[p_{r_2|\alpha,\beta}(R_2|\alpha,\beta)\right] &= \sum_{x=1}^{N}\sum_{y=1}^{N}\left[\ln\left(\frac{1}{\sqrt{2\pi}\sigma_n}\right)\right. \\ &\qquad \left. -\frac{(R_2(x,y)-R_1(x-\alpha,y-\beta))^2}{2\sigma_n^2}\right]. \end{aligned} \tag{A.13}$$

Thus the log-likelihood function is

$$L(\alpha,\beta)=\sum_{x=1}^{N}\sum_{y=1}^{N}\left[-\frac{(R_2(x,y)-R_1(x-\alpha,y-\beta))^2}{2\sigma_n^2}\right], \tag{A.14}$$

and Equation A.14 can be expanded into

$$\begin{aligned} L(\alpha,\beta) &= -\sum_{x=1}^{N}\sum_{y=1}^{N}\frac{R_2^2(x,y)}{2\sigma_n^2} \\ &\quad +2\sum_{x=1}^{N}\sum_{y=1}^{N}\frac{R_2(x,y)R_1(x-\alpha,y-\beta)}{2\sigma_n^2} \\ &\quad -\sum_{x=1}^{N}\sum_{y=1}^{N}\frac{R_1^2(x-\alpha,y-\beta)}{2\sigma_n^2}. \end{aligned} \tag{A.15}$$

By inspection of the terms in Equation A.15, it is clear that the first term is not affected by α and β. Thus, it does not affect the value of the log-likelihood summation. The third term is affected by the α and β shifts only around the edges of the image. At the edges pixel values could potentially vary significantly depending on the intensity variations between images along the edges. This could potentially add or subtract significantly in the summation. However, to simplify the analysis, it is assumed that the edge effects are negligible and the third term can be ignored. Hence, Equation A.15 simplifies to the middle cross-correlation term, which results in

$$L(\alpha,\beta)=\frac{1}{\sigma_n^2}\sum_{x=1}^{N}\sum_{y=1}^{N}\left[R_2(x,y)R_1(x-\alpha,y-\beta)\right]. \tag{A.16}$$

Simply put, the α and β shifts that create a peak value in the cross-correlation function are the estimators for α and β

$$(\hat{\alpha}, \hat{\beta}) = \underset{(\alpha,\beta)}{argmax} \Big[L(\alpha, \beta) \Big]. \tag{A.17}$$

Appendix B. Cross-Correlation vs. Vector-Block Performance Analysis

This appendix describes additional background research conducted on the two registration algorithms in [5] that produced both the fastest processing times and the lowest error metric (MSE): cross-correlation and vector-block. A self-study analysis was conducted on these two candidates, where simulations were performed to evaluate and compare their MSE and processing time performance metrics as in [5]. This appendix outlines the methodology of the simulations, provides results from the analysis, and considers conclusions about the mini-study.

B.1 Methodology

The simulation analysis for both registration algorithms is conducted using real-world ground test images generated by the AFRL Laservision system. Images of a resolution target truck at 3 km and a tank at 10 km are used to compare each MSE performance for short range and long range. The algorithms derived above are implemented in MATLAB® for use in the simulations. The next two sections present the results of the images processed by each algorithm at each range. Each section presents a baseline image of the scene as it appears without registration or temporal averaging (highlighting the speckle noise inherent to laser generated images), an image of the scene temporal averaged from 50 images, and an image with registration and temporal averaging of the 50 images. The results of the MSE and processing performance are then presented at the end of each section.

B.2 Simulation Results: 3 km Data

The 3 km test scene is of the resolution target truck, and Figure B.1 illustrates the baseline scene and temporal averaging without registration using 50 images. Note that the averaging process suppresses some noise, but the lack of registration, or aligning the images, causes severe blurring. Figure B.2 shows the visual improvements

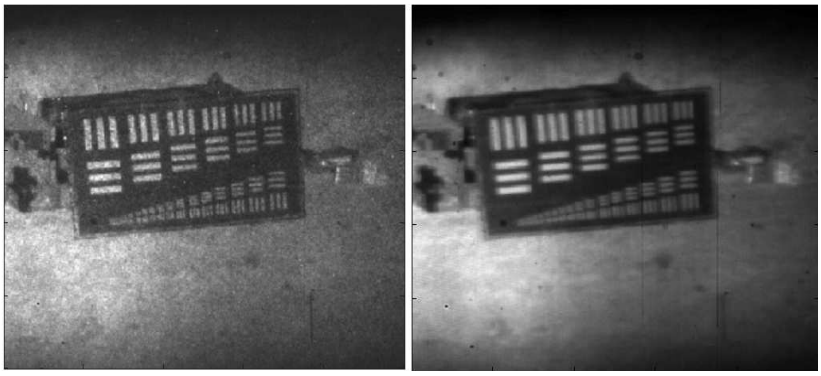


Figure B.1: **3 km Baseline Scene and Temporal Averaging without Registration.** These images illustrate the baseline scene with noise (left image) and temporal averaging without registration using 50 baseline scene images (right image). The speckle noise is somewhat suppressed, but there is severe blurring due to the lack of registration.

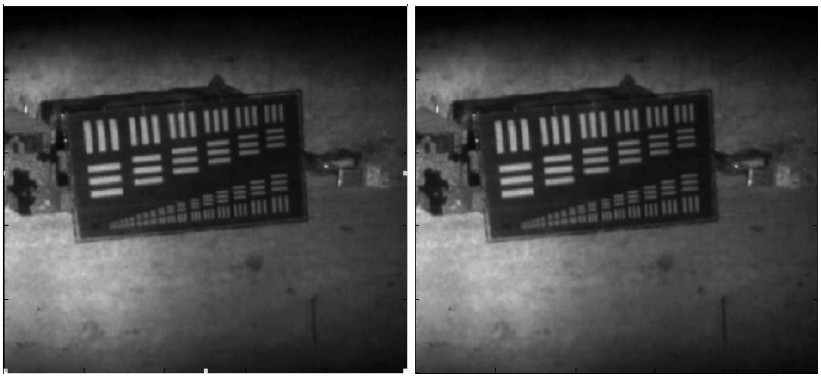
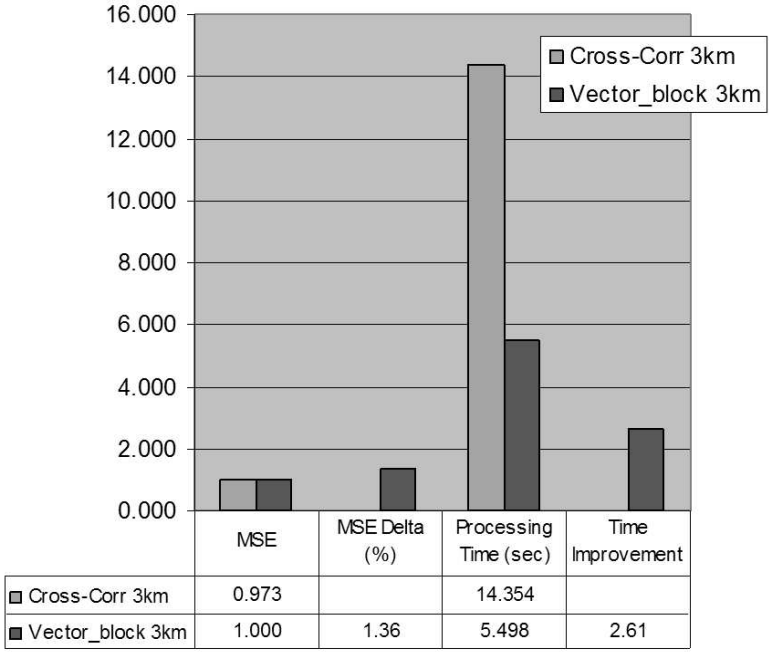


Figure B.2: **3 km Cross-correlation vs. Vector-Block Registration.** Illustrates the visual improvements of the noise blurring for each registration technique, cross-correlation on the left and vector-block on the right. Note there is very little difference seen with the eye. This result is confirmed by the MSE comparison, which shows a degradation in the vector-block method of only 1.36%.



	MSE	MSE Delta (%)	Processing Time (sec)	Time Improvement
Cross-Corr 3km	0.973		14.354	
Vector_block 3km	1.000	1.36	5.498	2.61

Figure B.3: **3 km Performance Metric Results, MSE and Processing Time.** The MSE comparison highlights a degradation of only 1.36% in the vector-block method versus the cross-correlation method. Furthermore, the most significant difference is that the vector-block method is over two-and-half times faster. Note: the MSE metric measurement is normalized to one in order to plot the two metrics on the same chart.

of the noise blurring for each registration technique, cross-correlation on the left and vector-block on the right.

Figure B.2 clearly demonstrates there is very little difference seen with the eye. This result is confirmed by the MSE comparison which shows a degradation in the vector-block method by only 1.36%, as depicted in Figure B.3. However, most striking is the computer processing time required for the vector-block method vs. the cross-correlation method, which, based on the P4 3GHz processor used, is over two-and-a-half times faster. Note: the MSE metric measurement is normalized to one in order to plot the two metrics on the same chart.

B.3 Simulation Results: 10 km Data

The 10 km test scene is of a tank, and Figure B.4 illustrates the baseline scene and temporal averaging without registration using 50 images. Note that for this longer range image the averaging process again suppresses the noise, and because the noise is more pronounced at this range the noise is suppressed enough to clear the image somewhat even in the absence of registration. Figure B.5 shows the visual improvements of the noise blurring for each registration technique, cross-correlation on the left and vector-block on the right. As in the 3 km test, Figure B.5 clearly demonstrates that there is very little difference seen with the eye. This result is again confirmed by the MSE comparison, which shows a degradation in the vector-block method by only 0.05%, as depicted in Figure B.6. However, most striking once again is the computer processing time required for the vector-block method vs. the cross-correlation method, which is still over two times faster. Note: the MSE metric measurement is normalized to one in order to plot the two metrics on the same chart.

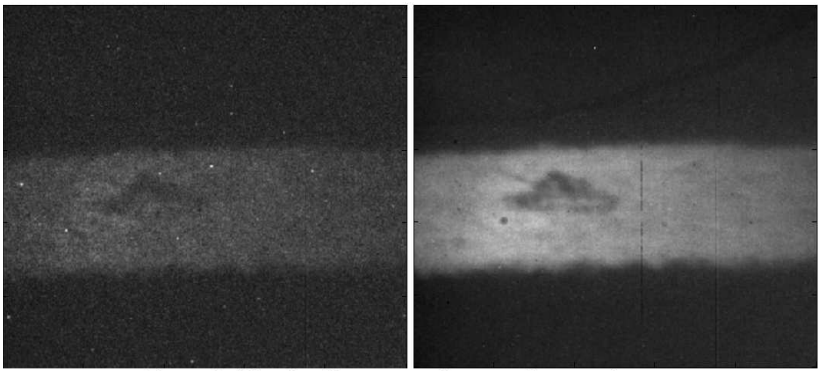


Figure B.4: **10 km Baseline Scene and Temporal Averaging without Registration.** These images illustrate the baseline scene with noise (left image) and temporal averaging without registration using 50 baseline scene images (right image). Because the noise is more pronounced at this range, the averaging process suppresses it enough to clear the image somewhat even in the absence of registration.

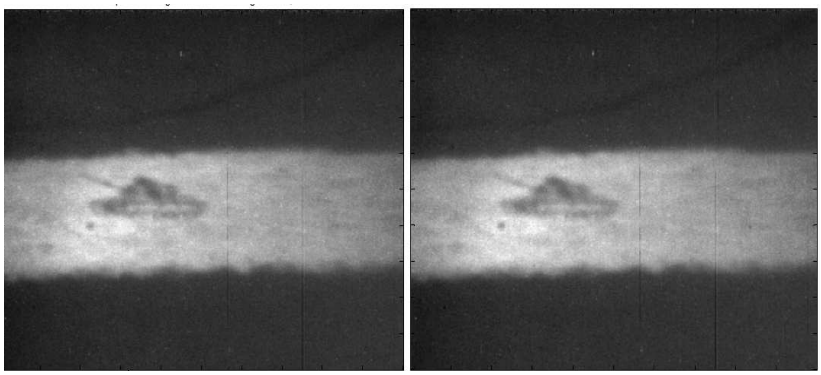
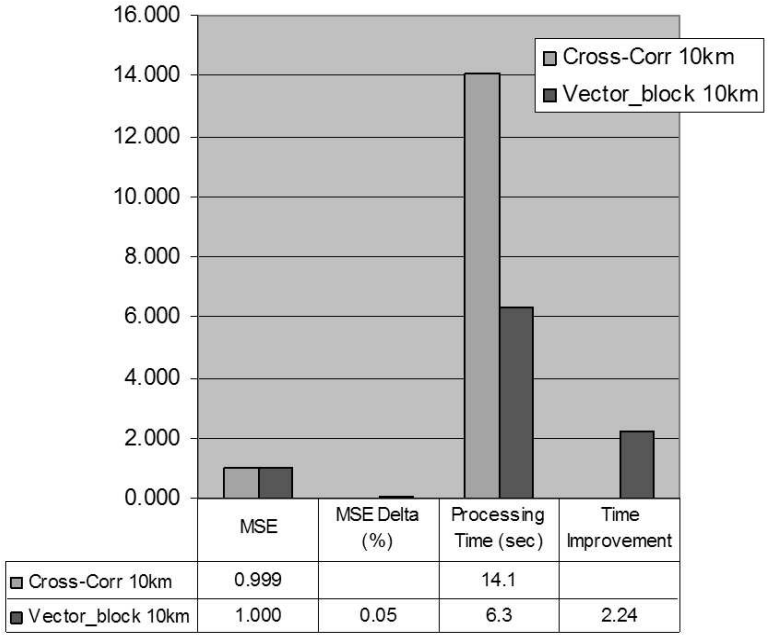


Figure B.5: **10 km Cross-correlation vs. Vector-Block Registration.** Illustrates the visual improvements of noise blurring for each registration technique, cross-correlation on the left and vector-block on the right. Note that there is very little difference seen with the eye. This result is confirmed by the MSE comparison, which shows a degradation in the vector-block method by only 0.05%.



	MSE	MSE Delta (%)	Processing Time (sec)	Time Improvement
Cross-Corr 10km	0.999		14.1	
Vector_block 10km	1.000	0.05	6.3	2.24

Figure B.6: **10 km Performance Metric Results, MSE and Processing Time.** The MSE comparison highlights a degradation of only 0.05% in the vector-block method versus the cross-correlation method. Furthermore, the most significant difference is that the vector-block method is over two times faster. Note: the MSE metric measurement is normalized to one in order to plot the two metrics on the same chart.

B.4 Mini-Study Conclusion

The most obvious conclusion drawn from this mini-study is the dramatic reduction in the cross-correlation computational load due to dimensionality reduction of the calculations, which results in striking differences in processing times between the two algorithms. Equally striking is how little variance there is in MSE, which translates into maintaining quality in the final image product. Coupled with the significant computer processing time saving, the vector-block method is an attractive substitute for the cross-correlation method. However, this result does not detract from potential improvement that can be gained through developing a registration algorithm using the true noise effect distribution statistics in the shift estimation process.

Appendix C. Computer Code

This appendix provides examples of the Matlab® computer code developed for both the simulated and real comparison tests conducted on the LITE and Cross-correlation algorithms.

Listing C.1: This first listing is the Matlab® code used for preparing the simulated data. It consists of using a true image scene (i.e., a scene with no noise), which is taken from the Laservision data base and generated through other research efforts. This true scene is replicated and each replicated scene is scaled to generated an average SNR value for the range of SNR values simulated (0.8 - 7).
(appendix3/list1SNR.m)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Major Darren R. Sabo
% Fall 2004
% AFIT/ENG Thesis Project
% Laser Image Registration Algorithm
% Average SNR range scaling of images
%

clear all;
close all;
clc

load M;

tic;
%--------------
% Set range of SNR values
%--------------

SNR_vals_1 = [0.2:.2:1];
SNR_vals_2 = [1.2:.2:2];
SNR_vals_3 = [2.2:.2:3];
SNR_vals_4 = [3.2:.2:4];
SNR_vals_5 = [4.2:.2:5];

N1 = size(SNR_vals_1,2);
N2 = size(SNR_vals_2,2);
N3 = size(SNR_vals_3,2);
N4 = size(SNR_vals_4,2);
N5 = size(SNR_vals_5,2);

%----------------
% Find mean roots for scaling images
%----------------

for i = 1:N1
    C1 = [(1-SNR_vals_1(i)/M) -SNR_vals_1(i) 0];
```

```
        mean_roots1(:,:,i) = roots(C1);
    end

    [i1,j1,v1] = find(mean_roots1);

    v1;

    for i = 1:N2
        C2 = [(1-SNR_vals_2(i)/M) -SNR_vals_2(i) 0];
        mean_roots2(:,:,i) = roots(C2);
    end

    [i2,j2,v2] = find(mean_roots2);

    v2;

    for i = 1:N3
        C3 = [(1-SNR_vals_3(i)/M) -SNR_vals_3(i) 0];
        mean_roots3(:,:,i) = roots(C3);
    end

    [i3,j3,v3] = find(mean_roots3);

    v3;

    for i = 1:N4
        C4 = [(1-SNR_vals_4(i)/M) -SNR_vals_4(i) 0];
        mean_roots4(:,:,i) = roots(C4);
    end

    [i4,j4,v4] = find(mean_roots4);

    v4;

    for i = 1:N5
        C5 = [(1-SNR_vals_5(i)/M) -SNR_vals_5(i) 0];
        mean_roots5(:,:,i) = roots(C5);
    end

    [i5,j5,v5] = find(mean_roots5);

    v5;

    %-----------------
    % Create scaling variables
    %-----------------
    load true2_center;
    true_mean = mean(mean(true2_center));

    for i = 1:length(v1)
        scale1(i) = v1(i)/true_mean;
    end
```

```
90
    for i = 1:length(v2)
        scale2(i) = v2(i)/true_mean;
    end

95  for i = 1:length(v3)
        scale3(i) = v3(i)/true_mean;
    end

    for i = 1:length(v4)
100     scale4(i) = v4(i)/true_mean;
    end

    for i = 1:length(v5)
        scale5(i) = v5(i)/true_mean;
105 end

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Scale images to vary SNR light intensities
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
110 for i = 1:length(v1)
        true_img_scaled1(:,:,i) = scale1(i).*true2_center;
    end

    for i = 1:length(v2)
115     true_img_scaled2(:,:,i) = scale2(i).*true2_center;
    end

    for i = 1:length(v3)
        true_img_scaled3(:,:,i) = scale3(i).*true2_center;
120 end

    for i = 1:length(v4)
        true_img_scaled4(:,:,i) = scale4(i).*true2_center;
    end
125
    for i = 1:length(v5)
        true_img_scaled5(:,:,i) = scale5(i).*true2_center;
    end

130 save('scale.mat','v1','v2','v3','v4','SNR_vals_1',...
        'SNR_vals_2','SNR_vals_3','SNR_vals_4',...
        'true_img_scaled1','true_img_scaled2','true_img_scaled3',...
        'true_img_scaled4','v5','SNR_vals_5','true_img_scaled5');

135 toc;
```

Listing C.2: This next listing is the Matlab® code used for processing the simulated data and running the comparison test between the LITE and Cross-correlation algorithms. The code is documented well enough to understand how the Monte Carlo simulation is executed.
(appendix3/list2simulated.m)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Major Darren R. Sabo
% Fall 2004
% AFIT/ENG Thesis Project
% Laser Image Registration Algorithm
% Monte Carlo Simulation
% CC vs. LITE algorithm comparison
% With varying SNR intensity values
% FILE #1_1
% Approx. time to run: 11 hrs w/ 20 SNR values
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% NOTE: The average SNR values (0.8 - 7.0) and their associated
% images that were scaled to those average values were saved in
% a separate .mat files and were broken up into several batch
% files for separate processing in order to run on multiple PCs
% to save time.
%
% This is one example of those batch files to illustrate how the
% simulations were set up and run.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;
close all hidden;
clc

%Resets it to a different state each time.
rand('state',sum(100*clock));

% Loads in the specific SNR values for this batch file.
S = load('scale.mat','SNR_vals_1');

SNR_vals_1 = S.SNR_vals_1;
clear S;

SNR_index = [4 5];

SNR_vals_1 = SNR_vals_1(SNR_index);

N=size(SNR_vals_1,2);

dB = 10*log10(SNR_vals_1);

N1 = 1000; % Number of noise realizations
M=50;
```

```
R=M;

tic;
for a = 1:N % SNR value loop.

    % Loads in the true image cube
    S = load('scale.mat','true_img_scaled1');

    % Pulls out only the image associated with the SNR value.
    true_img_scaled1 = S.true_img_scaled1(:,:,SNR_index(a));

    clear S;
    rows = size(true_img_scaled,1);
    cols = size(true_img_scaled,2);

    for i = 1:N1 % Noise realization loop.
        true_tank_cube1 = repmat(true_img_scaled1+1,[1 1 2]);

        N2 = size(true_tank_cube1,3);

        for b = 1:N2 % Loop to individually select each image and
                     % add random, independent negative binomial
                     % noise.

            K_bar1 = true_tank_cube1(:,:,b);

            % ------------------------
            % Use Matlab nbinrnd to generate negative binomial
            % distributed random numbers.
            % ------------------------

            P1 = (M./(M+K_bar1));

            RND_negbin_noise = nbinrnd(R,P1,500,500);
            tank_sim_wnoise_cube1(:,:,b) = RND_negbin_noise+1;

            clear RND_negbin_noise;
            clear K_bar1;
            clear P1;

            rand('state',sum(100*clock));
        end

%             figure;
%             imagesc(tank_sim_wnoise_cube1(:,:,1))
%             colormap(gray);

        clear true_tank_cube1;

        %---------------------------
        % Register with LITE algorithm
        %---------------------------
```

```
G1_1(:,:) = log(tank_sim_wnoise_cube1(:,:,1));
% The log of a ratio is the difference the logs of the
% numerator and denominator can be computed separately.
G2_1=log(tank_sim_wnoise_cube1(:,:,1)+M);

G1_1=conj(fft2(G1_1));
G2_1=conj(fft2(G2_1));

for k = 2:N2
    temp = fft2(tank_sim_wnoise_cube1(:,:,k));

    % This is the term produced by the numerator
    Cross1_LITE1 = real(ifft2(temp.*G1_1));

    % This is the term produced by the denominator
    Cross2_LITE1 = real(ifft2(temp.*G2_1));

    FCross_LITE1 = fftshift(Cross1_LITE1-Cross2_LITE1);

    clear Cross1_LITE1;
    clear Cross2_LITE1;
    clear G1_1;
    clear G2_1;

    % Find the maximum PSD point to determine
    % registration shift:
    [rowmax,colmax] = find(FCross_LITE1==max(max...
        (FCross_LITE1)));

    clear FCross_LITE1;

    shiftrow = rowmax-251;
    shiftcol = colmax-251;
    dx_LITE1(k-1)=-shiftcol;
    dy_LITE1(k-1)=-shiftrow;

end
dx_LITE_vec1_1(i) = dx_LITE1;
dy_LITE_vec1_1(i) = dy_LITE1;


%---------------------------
% Register with CC algorithm
%---------------------------

Baseline_cc1 = repmat(tank_sim_wnoise_cube1(:,:,1),...
    [1 1 N2]);

for m = 2:N2
    Cross_cc1 = real(ifft2(fft2(Baseline_cc1(:,:,m))...
        .*conj(fft2(tank_sim_wnoise_cube1(:,:,m)))));
```

```
150             FCross_cc1 = fftshift(Cross_cc1);

                clear Cross_cc1;

                % Find the maximum PSD point to determine
155             % registration shift:
                [rowmax,colmax] = find(FCross_cc1==...
                    max(max(FCross_cc1)));
                shiftrow = rowmax-251;
                shiftcol = colmax-251;
160             dx_cc1(m-1)=shiftcol;
                dy_cc1(m-1)=shiftrow;

                clear FCross_cc1;

165         end
            dx_cc_vec1_1(i) = dx_cc1;
            dy_cc_vec1_1(i) = dy_cc1;

            clear tank_sim_wnoise_cube1;
170         clear Baseline_cc1;

        end

        sigma_LITE1_1(a,:) = N^-0.5*sqrt((sum(dx_LITE_vec1_1.^2))...
175         + (sum(dy_LITE_vec1_1.^2)));

        sigma_cc1_1(a,:) = N^-0.5*sqrt((sum(dx_cc_vec1_1.^2))...
            + (sum(dy_cc_vec1_1.^2)));

180     toc;

    end
    toc;

185 sigma_LITE1_1
    sigma_cc1_1

    figure;
    plot(SNR_vals_1,sigma_LITE1_1,SNR_vals_1,sigma_cc1_1)
190 xlabel('SNR');
    ylabel('\sigma_{reg} (pixels)');
    legend('LITE','Cross-Corr');

    save('compare1_3_1000.mat','sigma_LITE1_1','sigma_cc1_1',...
195     'dx_LITE_vec1_1','dy_LITE_vec1_1','dx_cc_vec1_1',...
        'dy_cc_vec1_1','rowmax','colmax');
```

Listing C.3: This listing is the Matlab® code used for processing the real data and running the comparison test between the LITE and Cross-correlation algorithms. The code is documented well enough to understand how the algorithms are tested. (appendix3/list3real.m)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Major Darren R. Sabo
% Winter 2005
% AFIT/ENG Thesis Project
% Laser Image Registration Algorithm
%
% Real data comparison LITE vs. CC
% Standard Deviation Metric
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;
close all;
clc
load M;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------
% Load & view real resolution target data with speckle noise.
%-------------------------------------------------------------------
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
load restgt_data_cube;
N1 = size(data_cube,1); % 500
N2 = size(data_cube,2); % 500
N3 = size(data_cube,3); % 50 images

% Find the middle row and column values of image for use in the
% algorithms.
rows = 1:N1;
mid_row = round(median(rows));
cols = 1:N2;
mid_col = round(median(cols));

% figure;
% imagesc(data_cube(:,:,1))
% colormap(gray)
% set(gca,'FontSize',11);
% title('Real data w/ Speckle Noise');
% %axis off;


%---------------------------
% Register with LITE algorithm
%---------------------------
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate the LITE registration algorithm:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

restgt_LITE = zeros(N1,N2,N3); % Initializes the resolution
                               % target registration algorithm
                               % processing.

restgt_LITE(:,:,1)=data_cube(:,:,1); % Sets first image in the
                                     % restgt_LITE cube equal to
                                     % the first image of the
                                     % data cube file.

G1(:,:) = log(data_cube(:,:,1)+.001); % The log of a ratio is the
                                      % difference of their logs
                                      % so the logs of the
                                      % numerator and denominator
                                      % can be computed separately

G2=log(data_cube(:,:,1)+M);
G1=conj(fft2(G1));
G2=conj(fft2(G2));

tic;
for i = 2:N3
    temp=fft2(data_cube(:,:,i));

    % This is the term produced by the numerator
    Cross1 = real(ifft2(temp.*(G1)));

    % This is the term produced by the denominator
    Cross2 = real(ifft2(temp.*(G2)));

    FCross = fftshift(Cross1);

    % Find the maximum PSD point to determine registration shift:
    [rowmax,colmax] = find(FCross==max(max(FCross)));
    shiftrow = rowmax-mid_row; % finds row shift value
    shiftcol = colmax-mid_col; % finds column shift value
    dx_LITE(i)=-shiftcol;
    dy_LITE(i)=-shiftrow;

    restgt_LITE(:,:,i) = makeshift(data_cube(:,:,i),...
        dx_LITE(i),dy_LITE(i)); % makeshift is a separate
                                % Matlab routine that shifts each
                                % image in the cube by their
                                % respective row & column shifts
                                % estimated by the algorithm. The
                                % makeshift code is included in
                                % this Appendix.
end
```

```
LITE_TIME = toc;

clear Cross1 Cross2 FCross G1 G2;

%---------------------------
% Register with CC algorithm
%---------------------------

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Replicate image 1 from image data cube to create baseline image
% for registration
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Baseline_cc = data_cube(:,:,1);

Base_fft2 = fft2(Baseline_cc);

clear Baseline_cc;

tic;
for m = 1:N3
    Cross_cc = real(ifft2(Base_fft2.*...
        conj(fft2(data_cube(:,:,m)))));
    FCross_cc = fftshift(Cross_cc);

    % Find the maximum PSD point to determine registration shift:
    [rowmax,colmax] = find(FCross_cc==max(max(FCross_cc)));
    shiftrow = rowmax-mid_row; % finds row shift value
    shiftcol = colmax-mid_col; % finds column shift value
    dx_cc(m)=shiftcol;
    dy_cc(m)=shiftrow;

    restgt_cc_reg(:,:,m) = makeshift(data_cube(:,:,m),...
        dx_cc(m),dy_cc(m)); % makeshift is a separate
                            % Matlab routine that shifts each
                            % image in the cube by their
                            % respective row & column shifts
                            % estimated by the algorithm. The
                            % makeshift code is included in
                            % this Appendix.
end

CC_TIME = toc;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Temporal average unregistered images:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Temp_ave_unreg = mean(data_cube,3);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Temporal average registered images:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
Temp_ave_cc = mean(restgt_cc_reg,3);
Temp_ave_LITE = mean(restgt_LITE,3);

clear Cross1 Cross2 Cross_cc FCross FCross_cc G1 G2;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Display shifts and algorithm times
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
dy_LITE;
dy_cc;
dx_LITE;
dx_cc;
LITE_TIME;
CC_TIME;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Standard Deviation Comparison
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate Standard Deviation for LITE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

I_LITE = Temp_ave_LITE;

stdev_LITE=0;
for k = 1:N3
    stdev_LITE = stdev_LITE + sum(sum((restgt_LITE(:,:,k) -...
        I_LITE(:,:)).^2))/(N1*N2*N3);
end

clear I_LITE;

stdev_LITE = sqrt(stdev_LITE)


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate Standard Deviation for CC
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

I_CC = Temp_ave_cc;

stdev_CC=0;
for k = 1:N3
    stdev_CC = stdev_CC + sum(sum((restgt_cc_reg(:,:,k) -...
        I_CC(:,:)).^2))/(N1*N2*N3);
end

clear restgt_cc_reg I_CC;

stdev_CC = sqrt(stdev_CC)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate & Display Average SNR of scene
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Ave_data_cube = mean(mean(data_cube(:,:,1)))

Ave_SNR_data_cube = Ave_data_cube/stdev_LITE


save('Real_compare_021805.mat','dy_LITE','dx_LITE',...
    'LITE_TIME','dy_cc','dx_cc','CC_TIME','Ave_SNR_data_cube',...
    'stdev_LITE','stdev_CC');


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot images
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure;
subplot(221)
imagesc(data_cube(:,:,1))
colormap(gray)
set(gca,'FontSize',11);
title(['Real Scene, Ave SNR = ',...
    num2str(Ave_SNR_data_cube,'%3.2f')]);

subplot(222)
imagesc(Temp_ave_unreg);
colormap(gray);
axis off;
set(gca,'FontSize',11);
title(['Temp. Ave. Unreg., # Images = ', num2str(N3)]);

subplot(223)
imagesc(Temp_ave_LITE);
colormap(gray);
axis off;
set(gca,'FontSize',11);
title(['LITE, \sigma = ', num2str(stdev_LITE,'%3.2f')]);

subplot(224)
imagesc(Temp_ave_cc);
colormap(gray);
axis off;
set(gca,'FontSize',11);
title(['Cross-Correlation, \sigma = ',...
    num2str(stdev_CC,'%3.2f')]);

clear Temp_ave_LITE Temp_ave_cc Temp_ave_unreg;

%%%%%%%%%%%%%%%%%%%%    ZOOM1 START    %%%%%%%%%%%%%%%%%%%%%%%%%%%%
%=========================================
%=========================================
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Zoom on image to find lower SNR average scene
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
row1_z1 = 143;
row2_z1 = row1_z1+41;
col1_z1 = 201;
col2_z1 = col1_z1+41;

img_zoom1 = restgt_LITE(row1_z1:row2_z1,col1_z1:col2_z1,1);

mean_img_zoom1 = mean(restgt_LITE...
    (row1_z1:row2_z1,col1_z1:col2_z1,:),3);

N1z = size(mean_img_zoom1,1);
N2z = size(mean_img_zoom1,2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate & Display Average SNR of zoom scene
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
totimg_zoom1 = zeros(N1z,N2z);

mean_ave_zoom1 = mean(mean(mean_img_zoom1));

for i = 1:N3
 totimg_zoom1 = totimg_zoom1+(restgt_LITE...
     (row1_z1:row2_z1,col1_z1:col2_z1,i)-mean_img_zoom1).^2;
end

stdev_zoom1 = sqrt(sum(sum(totimg_zoom1))/(N3*N2z*N1z));

SNR_avg_zoom1 = mean_ave_zoom1/stdev_zoom1

%%%%%%%%%%%%%%%%%%%%%%%%    ZOOM1    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% View zoom scene
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% figure;
% imagesc(img_zoom1)
% colormap(gray)
%axis off;


%%%%%%%%%%%%%%%%%%%%%%%%    ZOOM1    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create zoom data cube:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
data_cube_zoom1 = data_cube(row1_z1:row2_z1,col1_z1:col2_z1,:);

N1z = size(data_cube_zoom1,1);
N2z = size(data_cube_zoom1,2);
N3z = size(data_cube_zoom1,3);

rows_zoom = 1:N1z;
```

```
mid_row_zoom = round(median(rows_zoom));
cols_zoom = 1:N2z;
mid_col_zoom = round(median(cols_zoom));

%=========================================
%=========================================
% Compare zoom scene
%=========================================
%=========================================

%---------------------------
% Register with LITE algorithm
%---------------------------

%%%%%%%%%%%%%%%%%%%%%%%%    ZOOM1    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate the LITE registration algorithm:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

restgt_LITE_zoom1 = zeros(N1z,N2z,N3z);
restgt_LITE_zoom1(:,:,1)=data_cube_zoom1(:,:,1);

G1(:,:) = log(data_cube_zoom1(:,:,1)+.001);
    % The log of a ratio is the difference of their logs so
    % the logs of the numerator and denominator can be
    % computed separately

G2=log(data_cube_zoom1(:,:,1)+M);
G1=conj(fft2(G1));
G2=conj(fft2(G2));

tic;
for i = 2:N3z
    %i;
    %pause(.1)
    temp=fft2(data_cube_zoom1(:,:,i));

    % This is the term produced by the numerator
    Cross1 = real(ifft2(temp.*(G1)));

    % This is the term produced by the denominator
    Cross2 = real(ifft2(temp.*(G2)));

    FCross = fftshift(Cross1-Cross2);

    % Find the maximum PSD point to determine registration shift:
    [rowmax,colmax] = find(FCross==max(max(FCross)));
    shiftrow = rowmax-mid_row_zoom;
    shiftcol = colmax-mid_col_zoom;
    dx_LITE_zoom1(i)=-shiftcol;
    dy_LITE_zoom1(i)=-shiftrow;

    restgt_LITE_zoom1(:,:,i) = makeshift(data_cube_zoom1...
```

```
        (:,:,i),dx_LITE_zoom1(i),dy_LITE_zoom1(i));
end

LITE_zoom1_TIME = toc;


%---------------------------
% Register with CC algorithm
%---------------------------

%%%%%%%%%%%%%%%%%%%%%%%%    ZOOM1    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Replicate image 1 from image data cube to create baseline image
% for registration
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Baseline_CC_zoom1 = data_cube_zoom1(:,:,1);

Base_fft2 = fft2(Baseline_CC_zoom1);

tic;
for m = 1:N3z
    Cross_cc = real(ifft2(Base_fft2.*...
        conj(fft2(data_cube_zoom1(:,:,m)))));

    FCross_cc = fftshift(Cross_cc);

    % Find the maximum PSD point to determine registration shift:
    [rowmax,colmax] = find(FCross_cc==max(max(FCross_cc)));
    shiftrow = rowmax-mid_row_zoom;
    shiftcol = colmax-mid_col_zoom;
    dx_CC_zoom1(m)=shiftcol;
    dy_CC_zoom1(m)=shiftrow;

    restgt_CC_zoom1_reg(:,:,m) = makeshift(data_cube_zoom1...
        (:,:,m),dx_CC_zoom1(m),dy_CC_zoom1(m));
end

CC_zoom1_TIME = toc;

%%%%%%%%%%%%%%%%%%%%%%%%    ZOOM1    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Temporal average unregistered images:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Temp_ave_unreg_zoom1 = mean(data_cube_zoom1,3);


%%%%%%%%%%%%%%%%%%%%%%%%    ZOOM1    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Temporal average registered images:
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Temp_ave_CC_zoom1 = mean(restgt_CC_zoom1_reg,3);
Temp_ave_LITE_zoom1 = mean(restgt_LITE_zoom1,3);

clear Cross1 Cross2 Cross_cc FCross FCross_cc G1 G2;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%    ZOOM1    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Display shifts and algorithm times
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
dy_LITE_zoom1;
dy_CC_zoom1;
dx_LITE_zoom1;
dx_CC_zoom1;
LITE_zoom1_TIME
CC_zoom1_TIME

%%%%%%%%%%%%%%%%%%%%%%%%    ZOOM1    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Standard Deviation Comparison
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%    ZOOM1    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate Standard Deviation for LITE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

I_LITE_zoom1 = Temp_ave_LITE_zoom1;

stdev_LITE_zoom1=0;
for k = 1:N3z
    stdev_LITE_zoom1 = stdev_LITE_zoom1 ...
        + sum(sum((restgt_LITE_zoom1(:,:,k) ...
        - I_LITE_zoom1(:,:)).^2))/(N1z*N2z*N3z);
end

stdev_LITE_zoom1 = sqrt(stdev_LITE_zoom1)

%%%%%%%%%%%%%%%%%%%%%%%%    ZOOM1    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate Standard Deviation for CC
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

I_CC_zoom1 = Temp_ave_CC_zoom1;

stdev_CC_zoom1=0;
for k = 1:N3z
    stdev_CC_zoom1 = stdev_CC_zoom1 ...
        + sum(sum((restgt_CC_zoom1_reg(:,:,k) ...
        - I_CC_zoom1(:,:)).^2))/(N1z*N2z*N3z);
end

stdev_CC_zoom1 = sqrt(stdev_CC_zoom1)



%%%%%%%%%%%%%%%%%%%%%%%%    ZOOM1    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot images
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure;
```

```
    subplot(221)
    imagesc(data_cube_zoom1(:,:,1))
    colormap(gray)
    set(gca,'FontSize',11);
    title(['Zoomed Scene, Ave SNR = ', ...
        num2str(SNR_avg_zoom1,'%3.2f')]);

    subplot(222)
    imagesc(Temp_ave_unreg_zoom1);
    colormap(gray);
    axis off;
    set(gca,'FontSize',11);
    title(['Temp. Ave. Unreg., # Images = ', num2str(N3)]);

    subplot(223)
    imagesc(Temp_ave_LITE_zoom1);
    colormap(gray);
    axis off;
    set(gca,'FontSize',11);
    title('Temp. Ave. LITE Reg.');
    title(['LITE, \sigma = ', num2str(stdev_LITE_zoom1,'%3.2f')]);

    subplot(224)
    imagesc(Temp_ave_CC_zoom1);
    colormap(gray);
    axis off;
    set(gca,'FontSize',11);
    title(['Cross-Correlation, \sigma = ', ...
        num2str(stdev_CC_zoom1,'%3.2f')]);

    save('Real_compare_zoom1_021905.mat','dy_LITE_zoom1',...
        'dx_LITE_zoom1','LITE_zoom1_TIME','dy_CC_zoom1',...
        'dx_CC_zoom1','CC_zoom1_TIME','SNR_avg_zoom1',...
        'stdev_LITE_zoom1','stdev_CC_zoom1');


    %%%%%%%%%%%%%%%%%%%    ZOOM2 START   %%%%%%%%%%%%%%%%%%%%%%%%%
    %=========================================
    %=========================================
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % NOTE: THIS ZOOM-IN PROCESS WAS REPEATED FOR OTHER SUB-REGIONS
    % OF THE LARGER SCENE TO FIND OTHER LOWER AVERAGE SNR AREAS TO
    % TEST THE TWO ALGORITHMS AGAINST.
    %
    % A TOTAL OF 4 SMALLER REGIONS WAS FOUND WITH AVERAGE SNRs BETWEEN
    % 4.17 AND 5.15.
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %=========================================
    %=========================================


    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% A total of the range values and the standard deviations is
% compiled to plot.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

SNR_range = [SNR_avg_zoom3 SNR_avg_zoom4 SNR_avg_zoom2 ...
    SNR_avg_zoom1 Ave_SNR_data_cube]

Stdev_LITE_rg = [stdev_LITE_zoom3 stdev_LITE_zoom4 ...
    stdev_LITE_zoom2 stdev_LITE_zoom1 stdev_LITE]

Stdev_CC_rg = [stdev_CC_zoom3 stdev_CC_zoom4 stdev_CC_zoom2...
    stdev_CC_zoom1 stdev_CC]

figure;
plot(SNR_range,Stdev_LITE_rg,'-o','LineWidth',1.5,'MarkerSize',5);
hold on;
plot(SNR_range,Stdev_CC_rg,':xr','LineWidth',1.5,'MarkerSize',7)
%axis tight;
set(gca,'FontSize',11);
xlabel('SNR');
ylabel('\sigma_{i} (pixel intensity)');
legend('LITE','Cross-Corr');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% A very simple processing time calculation analysis is done
% using the tic/toc commands seen above to provide an estimated
% percentage difference between the two algorithms.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Perc_improve_full = 100*(1-LITE_TIME/CC_TIME)

LITE_ztimes_ave = mean([LITE_zoom3_TIME LITE_zoom4_TIME ...
    LITE_zoom2_TIME LITE_zoom1_TIME])

CC_ztimes_ave = mean([CC_zoom3_TIME CC_zoom4_TIME ...
    CC_zoom2_TIME CC_zoom1_TIME])

Perc_improve_zoom = 100*(1-LITE_ztimes_ave/CC_ztimes_ave)

save('Real_compare_total_022005.mat','SNR_range',...
    'Stdev_LITE_rg','Stdev_CC_rg','Perc_improve_full',...
    'LITE_ztimes_ave','CC_ztimes_ave','Perc_improve_zoom');
```

Listing C.4: This final listing is the Matlab® code for the simple shift registration routine provided for this research effort. The code is documented well enough to understand how it functions.
(appendix3/list4makeshift.m)

```
function [img2]= makeshift(img1,dx,dy)
% function [img2]= makeshift(img1,dx,dy)
% dy and dx are the shifts in the vertical and horizontal
% directions respectively img1 and img2 are the two images
% from a sequence of video delta is the denominator of the
% fraction of a pixel to which the estimation is to be done.
%
% Ex 1/10 pixel estimation means delta =10

sz=size(img1);
sz=max(max(sz));
center = [floor(sz/(2))+1];
                linx = -center+1:1:center-2;
                linx = -2*pi*linx/sz;
      linx = fftshift(linx);
      liny=linx';

px = cos(linx*dx)+sqrt(-1)*sin(linx*dx);
py = cos(liny*dy)+sqrt(-1)*sin(liny*dy);
img2 = real(ifft2(fft2(img1).*(py*px)));
```

Bibliography

1. Brown, Lisa Gottesfeld. "A survey of image registration techniques". *ACM Computing Surveys*, 24(4):325–376, December 1992.

2. Cain, S.C., M.M. Hayat, and E.A. Armstrong. "Projection-based image registration in the presence of fixed-pattern noise". *IEEE Transactions on Image Processing*, vol. 10, Iss. 12:1860–1872, December 2001.

3. Curtis, K., B. Jacobson, D. Severance, A.T. Snyder, and D. Graves. "Johann Kepler: Letting the Heavens Declare the Glory of God". *Christian History Institute: Glimpses*, Issue 67, 2003.

4. Goodman, Joseph W. *Statistical Optics*. John Wiley and Sons, Inc., New York, New York. A Wiley-Interscience Publication.

5. MacDonald, Adam. "Comparison of Registration Techniques for Speckle Suppression in 2D LADAR Image Sequences". *Proceedings of SPIE*, volume 5558, 202–213. 2004.

6. Manfra, Jennifer L. *Translation and Rotation Invariant Multiscale Image Registration*. Master's thesis, Graduate School of Engineering, Air Force Institute of Technology (AETC), Wright-Patterson AFB OH, March 2002. AFIT/GE/ENG/02M-16.

7. Robinson, D. and P. Milanfar. "Comparison of Registration Techniques for Speckle Suppression in 2D LADAR Image Sequences". *IEEE Proceedings of the 2003 International Conference on Image Processing*, volume 2, II – 323–6. 2003.

8. Van Trees, H. L. *Detection, Estimation and Modulation Theory*. John Wiley and Sons, Inc., New York, New York, 1968.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
21–03–2005	Master's Thesis	Sept 2004 — Mar 2005

4. TITLE AND SUBTITLE

DEVELOPMENT OF A ROBUST OPTICAL IMAGE REGISTRATION ALGORITHM FOR NEGATING SPECKLE NOISE EFFECTS IN COHERENT IMAGES GENERATED BY A LASER IMAGING SYSTEM

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Darren R. Sabo, Major, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Bldg 641
Wright-Patterson AFB, OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GE/ENG/05-17

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Robert M. Zumrick
AFRL/SNJT, Bldg 620
2241 Avionics Circle
Wright-Patterson AFB, OH 45433
Comm (937) 255-9902 x4380, DSN 785-9902 x4380

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT The Air Force Research Laboratory (AFRL) Sensors Directorate has constructed and tested a coherent LIght Detection And Ranging (LIDAR) imaging system called Laservision. Registration of individual images remains a significant problem in the generation of useful images collected using coherent imaging systems. Coherent images typically contain significant speckle noise created by the coherency of the laser. Each image collected by the system must be properly registered to allow for averaging the images to produce a single image with adequate resolution to allow detection and identification algorithms to operate accurately or for system operators to perform target detection and identification within a scene. An investigation of the performance of a new image registration algorithm designed using laser speckle noise statistics is conducted on data collected from the Laservision system. This thesis documents the design and performance of the proposed technique compared to that of a standard cross-correlation algorithm. Based on using only speckle noise statistics, the simulated data test results indicate that there is a small range of low average signal-to-noise ratios (SNR) where there is the potential to improve the shift estimation error by 0.1 to 0.16 pixels.

15. SUBJECT TERMS

SIGNAL PROCESSING, IMAGE PROCESSING, IMAGE REGISTRATION, IMAGE MOTION COMPENSATION, LADAR(LASER DETECTION AND RANGING), LIDAR, OPTICAL RADAR

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Stephen C. Cain, PhD
U	U	U	UU	95	19b. TELEPHONE NUMBER *(include area code)* (937) 255–6565, ext 4408; stephen.cain@afit.edu